Game Semantics for a Polymorphic Programming Language

J. LAIRD, University of Bath

This article presents a game semantics for higher-rank polymorphism, leading to a new model of the calculus System F, and a programming language which extends it with mutable variables. In contrast to previous game models of polymorphism, it is quite concrete, extending existing categories of games by a simple development of the notion of *question/answer labelling* and the associated *bracketing condition* to represent "copycat links" between positive and negative occurrences of type variables. Some well-known System F encodings of type constructors correspond in our model to simple constructions on games, such as the lifted sum.

We characterize the *generic* types of our model (those for which instantiation reflects denotational equivalence), and show how to construct an interpretation in which all types are generic. We show how mutable variables (à la Scheme) may be interpreted in our model, allowing the definition of polymorphic objects with local state. By proving definability of finitary elements in this model using a decomposition argument, we establish a full abstraction result.

Categories and Subject Descriptors: D.3.3 [**Programming Languages**]: Language Constructs and Features—*Polymorphism*; F.3.2 [**Logics and Meanings of Programs**]: Semantics of Programming Languages—*Denotational sematics*

General Terms: Languages, Theory

Additional Key Words and Phrases: Polymorphism, System F, references, game semantics, genericity, full abstraction

ACM Reference Format:
Laird, J. 2013. Game semantics for a polymorphic programming language. J. ACM 60, 4, Article 29 (August 2013), 27 pages.
DOI: http://dx.doi.org/10.1145/2505986

1. INTRODUCTION

Polymorphism is an important principle in programming languages, occurring in many and different contexts. *Game semantics* has emerged as a means of describing higher-order program behaviour accurately, also in a wide variety of settings. The aim of this article is to take further steps towards a semantic understanding of polymorphism via games by describing a model of a small but expressive programming language (with functions, higher-order state and full higher-rank polymorphism) which is simple and concrete (and effectively presentable) and which precisely captures parametricity through properties of genericity, definability and full abstraction.

Semantic ideas and methods have featured extensively in attempts to capture the essence of parametric polymorphism in a higher-order setting, (contributing, e.g., the notion of relational parametricity [Reynolds 1983]). It is often studied in the purely functional setting of the second-order λ-calculus, System F [Girard 1972; Reynolds 1974]. Whilst the economy of this typing system is particularly elegant, focussing on the pure calculus raises several issues. Many key examples of polymorphism (such as

This work was supported by the UK EPSRC under grant EP/HO23097.
Author's address: J. Laird, Department of Computer Science, University of Bath, Bath, UK; email: jiml@cs.bath.ac.uk.

0004-5411/2013/08-ART29 $15.00
DOI: http://dx.doi.org/10.1145/2505986

generic mutable objects) require side-effects such as recursion and local state which have significant semantic implications (see, e.g., Møgelberg and Simpson [2009]) for any model of System F typing. There are also tensions between representing proofs of second-order logic and polymorphic programs: should a model reflect an appropriate notion of proof equality, or stronger notions of equivalence such as genericity [Longo et al. 1993]? Finally, there is the question of completeness—inhabitation of System F types (i.e., provability in second-order logic) is not decidable and so the desirable properties of *full completess* and *effective presentability* are not consistent in this case.

We focus on polymorphism as a property of programming languages, and develop a semantics for a prototypical example—an extension of System F in which variables may be updated, and so correspond to locally declared references. This may be used as a basis for describing polymorphic objects with internal state (for example). Working with an operational semantics, which gives a notion of observation and thus an associated equivalence, provides a simple criterion for determining whether the model accurately reflects the uniform nature of polymorphic programs—namely *full abstraction*.

A further difficulty involved in modelling higher-rank polymorphism is that of *size*. This is not simply the formal problem of capturing the impredicativity of second-order quantification, but the fact that the solutions still typically involve directly specifying behaviour at every possible type instantiation, making direct model-checking, for example, infeasible. Our semantics takes a rather direct approach to modelling universal quantification on games based on refining the *well-bracketing* condition (and associated question and answer labelling) already used to impose local control flow in game models.

The main idea making this possible is based on the observation, evident in earlier game models of, for example, propositional variables in proofs [Abramsky and Jagadeesan 1994], that polymorphic proofs or programs correspond to *copycat strategies*: since such programs must behave uniformly over all instantiating types, all they can do is copy information between positive and negative occurrences. Thus, any representation of such a program which permits these "copycat links" to be inferred is sufficient to determine its behaviour on any type. The key development here is to show that they can be reconstructed using the well-bracketing condition: answering the pending question in a game denoted by a polymorphic type corresponds to playing a copycat link between any occurrences of any type instantiated for the variables represented by the question and its answer.

1.1. Related Work

Two game models of higher-order polymorphism, in the setting of System F, have been described previously. The first, due to Dominic Hughes [Hughes 1997], foreshadows our semantics in several particulars: it introduces a basic notion of polymorphic arena, giving a very similar interpretation of variable types. Higher rank polymorphism, however, is interpreted using a notion of *hypergame* allowing participants to import arenas instantiating variables, and changing the shape of interaction. This results in a model which closely reflects the syntax of System F—as shown by a full completeness result associating strategies to unique η-long normal forms [Hughes 1997], and recent applications in determining the class of *dinatural* terms of System F [de Lataillade 2009]. Our semantics could be said to describe polymorphism in a more programming language oriented way.

A subsequent game model of System F polymorphism was described by Samson Abramsky and Radha Jagadeesan [Abramsky and Jagadeesan 2004]. This work is focussed on developing a semantics capturing *genericity*, as expressed by the genericity principle for System F introduced by Longo et al. [1993]: a System F type T is generic in a given model if instantiation with T reflects denotational equivalence (i.e.,

any two terms of universally quantified type which are denotationally equivalent when instantiated with T are themselves denotationally equivalent). A semantics satisfies the genericity property if every type is generic. This captures the inability of polymorphic programs to exhibit different behaviour on different instantiating types in a strong sense—although it is consistent with the theory of System F, Longo et al. [1993] note that none of the models considered in that article satisfy the genericity principle: to the author's knowledge, no models which do so have been described until now. (The model in Abramsky and Jagadeesan [2004], although not, in fact, satisfying the genericity principle, contains a large collection of generic types.) In our semantics there is a simple criterion identifying generic types, and a simple modification making all types generic.

In other respects, our model is rather different from the semantics described in Abramsky and Jagadeesan [2004]; in particular, the latter interprets universal quantification as a dependent product over all games, requiring a model defined as the solution to a recursive equation, whereas the interpretations of types in our model are built up in a simple inductive fashion, with the encodings of constructions such as the lifted sum having equivalent denotations to their direct interpretations.

The interpretation of general references builds on the work of Abramsky et al. [1998], showing how general references may be interpreted by lifting the conditions of *innocence* and *visibility* placed on an original model of PCF, and further analysis of this model by the author [Laird 2002] identifying categorical structures which may be used to construct it.

This article is based on an extended abstract [Laird 2010a]. It describes the same results—a generic semantics of System F, and a fully abstract model of an extension with general references—in a different (but related) category of games. The aim of these revisions in presentation have been to present the key ideas of the model in a way which keeps the combinatorics of the game semantics as simple as possible, by avoiding the use of "justification pointers" and making use of the linear decomposition of intuitionistic types, and the categorical structures introduced in Laird [2002] to model references. The price paid is a little less generality (the model of System F described here does not include products, although it could be extended to include them) and a less direct relationship to the growing body of work on Hyland-Ong games. We refer back to [Laird 2010a] for indications of how these restrictions may be avoided.

1.2. Outline of this Article

In Section 2, we describe an extension of System F with mutable variables—System $\mathrm{F}_{\mathtt{ref}}$—and its operational semantics.

In Section 3, the fundamental definitions of the game semantics are introduced based on *context games* which have "holes" into which games may be instantiated, allowing the interpretation of types with free variables. In Section 4, a category of context games is defined using a new notion of abstraction on holes, together with a family of instantiation functors on it.

In Section 5, we show how intuitionistic function types may be interpreted by a construction of cofree commutative comonoids in our category, and a semantic representation of reference cells is described using further structure.

In Section 6, we give the interpretation of System $\mathrm{F}_{\mathtt{ref}}$ based on our constructions, and show that it is sound (and an instance of the Seely-Pitts hyperdoctrine interpretation of System F) based on the categorical properties of our game model. We give a simple proof of computational adequacy.

In Section 7, we give a simple characterization of the types which are generic in our semantics, and a modified construction of a model in which every type is generic. We prove that every finite element of our (original) model is denoted by a term, using

Table I. Term Formation Rules for System $\mathrm{F}_{\mathtt{ref}}$

$$\frac{\Theta \vdash \Gamma}{\Theta;\Gamma \vdash () : \mathsf{com}} \qquad \frac{\Theta \vdash \Gamma}{\Theta;\Gamma \vdash \Omega : \forall X.X}$$

$$\frac{\Theta \vdash \Gamma, T}{\Theta;\Gamma, x : T \vdash x : T} \qquad \frac{\Theta \vdash \Gamma, T}{\Theta;\Gamma, x : T \vdash \mathtt{set}(x) : T \to \mathsf{com}}$$

$$\frac{\Theta, X;\Gamma \vdash M : T}{\Theta;\Gamma \vdash \Lambda X.M : \forall X.T} \qquad \frac{\Theta;\Gamma \vdash M : \forall X.T \quad \Theta \vdash S}{\Theta;\Gamma \vdash M\{S\} : T[S/X]}$$

$$\frac{\Theta;\Gamma, x : S \vdash M : T}{\Theta;\Gamma \vdash \lambda x^S.M : S \to T} \qquad \frac{\Theta;\Gamma \vdash M : S \to T \quad \Theta;\Gamma \vdash N : S}{\Theta;\Gamma \vdash M\,N : T}$$

$$\frac{\Theta;\Gamma \vdash M : \mathsf{com} \quad \Theta;\Gamma \vdash N : T}{\Theta;\Gamma \vdash M;N : T}$$

an argument based on decomposition of strategies and prove a full abstraction result based upon it.

2. SYSTEM F WITH MUTABLE VARIABLES

We define a prototypical polymorphic programming language, System $\mathrm{F}_{\mathtt{ref}}$, by extending System F [Girard 1972; Reynolds 1974], with assignment to variables, in a similar style to the language `Scheme` (although evaluation is lazy).

Types of our language are defined as for System F, extended with a constant ground type of commands, com. This type is equivalent (isomorphic in the fully abstract model) to the pure System F type $\forall X.X \to X$. Including it as an explicit constant is a notational convenience which simplifies presentation of the operational semantics, and plays a role in the proof of full abstraction.

Type formation rules are as follows (here, and elsewhere, we use $X_0, \ldots, X_n$ as metavariables representing a context of actual type variables):

$$\frac{}{X_0,\ldots,X_n \vdash X_0} \qquad \frac{}{\Theta \vdash \mathsf{com}}$$

$$\frac{\Theta \vdash S \qquad \Theta \vdash T}{\Theta \vdash S \to T} \qquad \frac{\Theta, X \vdash T}{\Theta \vdash \forall X.T}$$

$$\frac{X_0,\ldots,X_n \vdash T}{X_{\pi(0)},\ldots,X_{\pi(n)} \vdash T}\ \pi \in perm\{0,\ldots,n\}$$

Term-formation rules and typing judgments, of the form $\Theta; x_1 : S_1, \ldots, x_n : S_n \vdash M : T$ where $\Theta \vdash S_1, \ldots, S_n, T$, are given in Table I. These are based on the rules for System F, extended with:

—A constant $()$: com (corresponding to $\Lambda X.\lambda x^X.x$).
—Sequential composition of M : com with any term.
—A constant Ω: $\forall X.X$ denoting error or non-termination.
—For each variable x: T, a term $\mathtt{set}(x) : T \to \mathsf{com}$. We may write $x := M$ for the assignment $\mathtt{set}(x)\,M$.

2.1. Operational Semantics

To give an operational semantics of System $\mathrm{F}_{\mathtt{ref}}$ we extend the syntax with an unbounded set of typed constants (*location names*) $a : T$ for each closed type T, which may take the place of variables. A *program* is a closed term over this extended language. A configuration $(M|\mathcal{E})$ is a pair of a program M : com and an environment $\mathcal{E}$—a finite set of pairs (a, N) representing a partial function from the set of location names to

Table II. Operational Semantics of System $\mathsf{F}_{\mathtt{ref}}$

$E[(\Lambda X.M)\{T\}]\|\mathcal{E}$	$\longrightarrow$	$E[M[T/X]]\|\mathcal{E}$
$E[(\lambda x.M)\ N]\|\mathcal{E}$	$\longrightarrow$	$E[M[a/x]]\|\mathcal{E}, (a, N)\ (a \notin \pi_l(\mathcal{E}))$
$E[(); M]\|\mathcal{E}$	$\longrightarrow$	$E[M]\|\mathcal{E}$
$E[\mathtt{set}(a)\ M]\|\mathcal{E}, (a, N)$	$\longrightarrow$	$E[()]\|\mathcal{E}, (a, M)$
$E[a]\|\mathcal{E}, (a, M)$	$\longrightarrow$	$E[M]\|\mathcal{E}, (a, M)$

the set of programs. The rules in Table II define a reduction relation on configurations based on the *evaluation contexts* given by the following grammar:

$$E[\_] ::= [\_] \mid E[\_\{T\}] \mid E[\_\ M] \mid E[\_; M]$$

Given a program $M : \mathsf{com}$, we write $M \Downarrow$ if $M|_- \longrightarrow ()|\mathcal{E}$ for some $\mathcal{E}$. We adopt standard definitions of contextual approximation and equivalence with respect to this notion of observation. Given terms $M, N : T$:

—$M \lesssim N$ if for all contexts $C[\_] : \mathsf{com}$, $C[M] \Downarrow$ implies $C[N] \Downarrow$.
—$M \approx N$ if $M \lesssim N$ and $N \lesssim M$.

2.2. Expressiveness of System $\mathsf{F}_{\mathtt{ref}}$

The combination of higher-rank polymorphism, mutable variables and lazy evaluation might be considered nonstandard from a programming language perspective, although the comparative simplicity of our model suggests that it is semantically natural (rather than being intrinsically related to call-by-name evaluation, our model describes polymorphism for *computation types* in the sense of Levy [2004]). The small set of syntactic operations suffice to define a number of powerful programming language constructs.

Data Types. Encodings of type constructors and inductive and abstract data types including the Booleans, lazy natural numbers, sum and list types as System F types are well known. In particular, we shall write $T_1 \& T_2$ for the product type $\forall X.(T_1 \to T_2 \to X) \to X$, where X is not free in T_1, T_2, with the pairing operation $\langle M_1, M_2 \rangle = \Lambda X.\lambda f^{T_1 \to T_2 \to X}.(f\, M_1)\, M_2$, and projections $\mathsf{fst} = \lambda x^{T_1 \& T_2}.x\{T_1\}\, \lambda y^{T_1}.\lambda z^{T_2}.y$ and $\mathsf{snd} = \lambda x^{T_1 \& T_2}.x\{T_2\}\, \lambda y^{T_1}.\lambda z^{T_2}.z$. (We write $T_\perp$ for the unary instance $\forall X.(T \to X) \to X$, with $\mathsf{proj} : T_\perp \to T = \lambda x^{T_\perp}.x\{T\}\, \lambda y^T.y$.)

Recursive Definitions. These may be expressed using self-referencing variables: for example, $\Lambda X.\lambda f^{X \to X}.(\mathsf{new}\, a^X.(a := f\, a); a) : \forall X.(X \to X) \to X$ defines a polymorphic fixpoint combinator (where $\mathsf{new}\, a^T.M = (\lambda a^T.M)\, \Omega\{T\}$ declares a fresh variable in M).

General References. We may represent the type of references of type T as the product type $\mathtt{var}[T] = T \& (T \to \mathsf{com})$ so that left-projection is dereferencing and right projection is assignment. The term $\mathtt{ref}_T : \lambda x^T.\langle x, \mathtt{set}(x) \rangle : T \to \mathtt{var}[T]$ declares a reference cell storing values of type T.

Polymorphic Mutable Objects. Extending the proposals in Abramsky et al. [1998], we may represent a *polymorphic* object as a term of type $\forall X_1 \ldots \forall X_m.T \to (S_1 \& \cdots \& S_n)$, where T is the type of an initial state and $S_1, \ldots, S_n$ are method types, parameterised over the variables $X_1, \ldots, X_m$. A simple example is the polymorphic reference cell in this article. Similarly, the term of type $\forall X.(X \& (X \to \mathsf{com}))$

$$\Lambda X.\mathsf{new}\, x^{X_\perp}.\langle \mathsf{proj}(x), \lambda y^X.x\{\mathsf{com}\}\, \lambda z^X.x := \langle (x := \langle z \rangle); y \rangle \rangle$$

represents a stack object (storing a list of terms of type X as a reference of type $X_\perp$ which returns a different value each time it is "unthunked"): `pop` is left projection and `push` is right-projection.

Dynamic Binding. We may use mutable variables of existential type ($\exists X.T =_{df} \forall Y.(\forall X.(T \to Y) \to Y)$ with Y not free in T) to store an object which instantiates

the quantified variable with any type—for example, we may dynamically bind a variable $a : \exists X.T$ to a term $M : T[S/X]$ by the assignment $a := \Lambda Y.\lambda f^{\forall X.(T \to Y)}.\, f\{S\}\, M$.

3. GAMES FOR POLYMORPHISM

Games are labelled forests of moves, similar to the "Hyland-Ong games" on which an earlier version of our semantics [Laird 2010a] is based. They are defined with respect to a fixed universe of *moves* (from which their forest structure is derived): the set $\mathcal{U} = \{0, 1\}^*$ of finite binary sequences. We write $\sqsubseteq$ for the prefix relation on sequences, and given any subset $X \subseteq \mathcal{U}$, we write X^- and X^+ for the subsets of X consisting of odd-length sequences ("Player moves") and even-length sequences ("Opponent moves") respectively.

From any partial function $f : \mathcal{U} \to \mathcal{U}$, we may derive a function on finite sequences over $\mathcal{U}$—$f^* : \mathcal{U}^* \to \mathcal{U}^*$ applies f pointwise to the moves on which it is defined, and omits moves on which it is not defined. Where a partial projection function p from $\mathcal{U}$ onto a subset $Y \subseteq \mathcal{U}$ is evident from the context, we shall write $s \restriction Y$ for $p^*(s)$—for example, the restriction of a sequence over a disjoint union $X_1 + \cdots + X_n$ to the disjoint union of a subset of the X_i. For any binary word u, $s \restriction u = p_u^*(s)$, where p_u is the partial function sending $u \cdot v$ to v—that is, $s \restriction u$ consists of the sequence of words which occur as suffixes of u in s.

3.1. Pre-Games

Define the pre-order $\preceq$ on $\mathcal{U}$ as follows:

$$m \preceq n \text{ if there exist } a, c \in \{0, 1\}^* \text{ and } b \in \{0\}^* \text{ such that } m = a \cdot b \text{ and } n = a \cdot c.$$

A *pre-game* A is a set of moves $M_A \subseteq \mathcal{U}$ such that $m, n \in M_A$ and $m \sqsubseteq n$ implies $m = n$.

PROPOSITION 3.1. *For any pre-game A, the restriction of $\preceq$ to M_A is a partial order.*

PROOF. Suppose $m \preceq_A n$ and $n \preceq_A m$. Then $m = a \cdot b = a' \cdot c'$ and $n = a' \cdot b' = a \cdot c$, where $b, b' \in \{0\}^*$. Suppose (without loss of generality) that $a \sqsubseteq a'$. Then, $c' \in \{0\}^*$, and so $b' \sqsubseteq c'$ or $c' \sqsubseteq b'$, and so $m \sqsubseteq n$ or $n \sqsubseteq m$ and hence $m = n$ as required. □

An initial move of A is a minimal element of $(M_A, \preceq)$. A pre-game is *well opened* if it contains a move $o \in \{0\}^*$, which must be the unique initial move.

All of the pre-games that we shall use correspond to regular languages over the alphabet $\{0, 1\}$, and so we may represent them using corresponding notation:

Prefixing. Given a sequence $u \in \mathcal{U}$, $u.A$ (the *offset* of A by u) is obtained by prefixing all moves in A with u.

Union. If A and B are pre-games such that $m \in A$ and $n \in B$ implies $m \not\sqsubseteq n$ and vice-versa, then their union is a pre-game. We write $C \uplus D$ for the disjoint union $10.C \cup 01.D$.

3.2. Context Games

Types with free variables will be interpreted by labelling moves with natural numbers representing explicit "holes", into which any other game can be plugged (similar to the polymorphic games of Hughes [1997], variable games of Abramsky and Jagadeesan [2004] and open games of Clairambault [2009]). Moves which are not holes are labelled as *questions* or *answers*.

Given any set $X \subseteq \mathbb{N}$, let $X_{QA} = \{Q, A\} \cup X$, and given $f : X \to Y$, let $f_{QA} : X_{QA} \to Y_{QA}$ extend f with the identity on $\{Q, A\}$. Formally, a (context) game A is a tuple $(M_A, \lambda_A, \rhd_A)$ such that:

—M_A is a pre-game.
—$\lambda_A : M_A \to \mathbb{N}_{QA}$ is a labelling function partitioning M_A into sets $\mathsf{Qn}_A = \lambda_A^{-1}\{Q\}$ (questions), $\mathsf{Ans}_A = \lambda_A^{-1}\{A\}$ (answers), and $H_A(i) = \lambda_A^{-1}\{i\}$ for each i ("i-holes").
—$\rhd_A \subseteq (\mathsf{Qn}_A^+ \times \mathsf{Ans}_A^-) \cup (\mathsf{Qn}_A^- \times \mathsf{Ans}_A^+)$ is an answer relation determining which answers respond to which questions.

We shall use the following terminology:

—A is *negative* if every initial move is an Opponent move, and is not an answer.
—A is *n-closed* if $H_A(n) = \varnothing$.
—A is a *n-context* game if it is m-closed for every $m \geq n$.
—A is *closed* if it is n-closed for all n—that is, a 0-context game.

Remark 3.2. The original notion of game in Hyland and Ong [2000] associates questions to their set of possible answers by an explicit function. In the reformulation of the notion of game by McCusker [1996], answers implicitly respond to their *enabling* question. Our notion of game, with an explicit answering *relation*, subsumes both of these definitions.

3.3. Well-Bracketing and Legal Sequences

The legal sequences of moves representing reachable *positions* of a game are simpler than Hyland-Ong games because no explicit "justification pointers" are required (their rôle is played by additional tags on moves induced by an explicit "bang" construction).

The set L_A of *legal sequences* of the closed game A consists of the finite, nonrepeating sequences s over M_A which satisfy the following conditions:

Alternation. s starts with an Opponent move (if nonempty) and alternates between Player and Opponent moves.

Downwards-closure. If $a \preceq_A b$, then a precedes b in s—that is, the moves in any prefix of s form a lower set.

Well-Bracketing. Every answer move is related to the last-asked, unanswered question by the $\rhd_A$ relation.

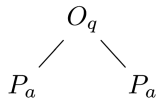
The well-bracketing condition is essentially as defined in Hyland and Ong [2000], and Abramsky et al. [2000] etcetera: we give further explanation as it will play a key role in our model. By definition, in a closed game, every move is either a question or an answer. A well-bracketed sequence on such a game corresponds to a sequence of opening and closing parentheses (questions and answers, respectively), such that each pair of opening and closing brackets are in the answering relation.

Formally, say that a sequence of moves $t \in M_A^*$ is *bracketed* if every prefix of t contains no more questions than answers and *complete* if t contains equal numbers of questions and answers. For any bracketed sequence s on M_A^*, we define a relation $\mathsf{Cl}_A(s) \subseteq \mathsf{Qn}_A \times \mathsf{Ans}_A$ as follows:

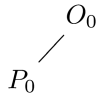
—$\mathsf{Cl}_A(\varepsilon) = \varnothing$,
—$\mathsf{Cl}_A(sq) = \mathsf{Cl}(s)$,
—$\mathsf{Cl}_A(sqs'a) = \mathsf{Cl}(s) \cup \mathsf{Cl}(s') \cup \{(q, a)\}$, if s' is complete.

s is a well-bracketed sequence on A if and only if $\mathsf{Cl}_A(s)$ is contained in $\rhd_A$.

Instantiating the game $\mathbb{B}$:



into the context game Σ:



gives

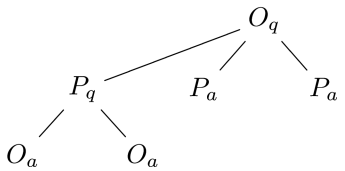


Fig. 1. Instantiation: An example.

3.4. Constructions on Context Games

Standard examples of closed games include Σ, which has a single Opponent question enabling a corresponding Player answer—that is, $M_\Sigma = \{00, 1\}$, $\lambda_\Sigma(00) = Q$, $\lambda_\Sigma(1) = A$ and $\rhd_\Sigma = \{(00, 1)\}$. Similarly, we have a game $\mathbb{B}$ of Booleans with a single Opponent question enabling two Player answers. We will show later how these are isomorphic to the interpretations of System F types in our model. The fundamental nontrivial examples of context games—corresponding to free type variables—are the (well-opened) games $\bullet_i$ for each i, containing the single (Opponent) move ε, which is an i-hole move—that is, $\lambda_{\bullet_i}(\varepsilon) = i$.

For any $X \subseteq \mathbb{N}$, and game A, the *universal quantification* of A over X, $\forall_X A$ is played over the same pre-game as A, with the Player n-hole moves becoming answers to the Opponent n-holes for each $n \in X$—that is, $\forall_X A = (M_A, \lambda_A[H_A^+(X) \to \{Q\}, H_A^-(X) \to \{A\}], \rhd_A \cup \bigcup\{H_A^+(n) \times H_A^-(n) \mid n \in X\})$.

3.5. Instantiation

We will interpret instantiation of type variables with types by substituting hole-moves with well-opened games. Given games A and B, where B is well opened, we may define the instantiation of B into A by replacing each 0-hole u with the moves $u \cdot B$—that is,

—$M_{A[B]} = M_A \backslash H_A(0) \cup \{m \cdot n \mid m \in H_A(0) \wedge n \in M_B\}$.
—$\lambda_{A[B]}(m) = \lambda_A(m)$, if $m \in M_A \backslash H_A(0)$,
$\lambda_{A[B]}(l \cdot m) = \lambda_B(m)$, if $l \in H_A(0)$.
—$\rhd_{A[B]} = \rhd_A \cup \{(m \cdot q, m \cdot a) \mid m \in H_A(0) \wedge (q, a) \in \rhd_B\}$. (See Figure 1)

4. A CATEGORY OF CONTEXT GAMES

A *strategy* on the closed game A is a nonempty, even-prefix-closed set of even-length legal sequences on A, which satisfies the determinacy condition: if $sa, sb \in \sigma$, then $a = b$.

Given negative games A, B, we define the (closed) game $A \to B$ to be $\forall_{\mathbb{N}}(1.A \cup 01.B)$—that is, the disjoint union of A and B with Player/Opponent polarity switched in A, with all hole moves quantified out (note that this is not, itself, a negative game in general). We shall define a category $\mathcal{G}$ in which objects are negative context games and morphisms from A to B are strategies on $A \to B$. To define notions of composition and identity on $\mathcal{G}$ (and the action of instantiation on strategies), we require the following notion of copycat sequence:

Definition 4.1. Given sets of moves $X, Y \subseteq \mathcal{U}$, a sequence s is a *Player* X, Y-copycat if for any *even-length* prefix $t \sqsubseteq s$, $t{\restriction}X = t{\restriction}Y$ and an Opponent X, Y-copycat if for any *odd-length* prefix $t \sqsubseteq s$, $t{\restriction}X = t{\restriction}Y$. (Given sequences $u, v \in \mathcal{U}$, a play is a (Player/Opponent) (u, v)-*copycat* if it is a U, V-copycat, where U, V are the sets of extensions of u and v.)

The identity morphism $\mathsf{id}_A : A \to A$ consists of legal sequences which are Player copycats between the positive and negative copies of A—that is, Player $(01, 1)$-copycat sequences on $A \to A = \forall_{\mathbb{N}}(1.A \cup 01.A)$.

Composition in $\mathcal{G}$ may be defined by a form of "parallel composition plus hiding" based on Opponent copycat sequences.

Definition 4.2. An *interaction sequence* on $A_1, \ldots, A_{n+1}$ is a sequence s on $(A_1 \to A_2) \uplus (A_2 \to A_3) \uplus \cdots \uplus (A_n \to A_{n+1})$ such that:

—s is an Opponent (A_i^-, A_i^+)-copycat for $2 \leq i \leq n$ (copycat condition).
—$s{\restriction}A_i \to A_{i+1}$ is legal for $1 \leq i \leq n$ (projection condition).

Using this notion, we define n-ary composition of strategies for $n \geq 2$ as follows:

Definition 4.3. Given $\sigma_1 : A_1 \to A_2, \ldots, \sigma_n : A_n \to A_{n+1}$, let the n-ary composition $\sigma_1 | \cdots | \sigma_n$ be the set of legal sequences s on $A_1 \to A_{n+1}$ such that there exists an interaction sequence t on $A_1, \ldots, A_{n+1}$ such that $t{\restriction}A_i \to A_{i+1} \in \sigma_i$ for $1 \leq i \leq n$ and $s = t{\restriction}A_1 \to A_{n+1}$.

LEMMA 4.4. *$\sigma_1 | \cdots | \sigma_n$ is a well-defined strategy.*

PROOF. $\sigma_1 | \cdots | \sigma_n$ is a set of legal sequences by definition. It consists of even sequences, since if t is an interaction sequence for which each projection $t{\restriction}A_i \to A_{i+1}$ is even-length, then t is even length, and the restriction of t to $A_1 \to A_{n+1}$ excludes an even number of moves by the copycat condition. Even-prefix closure is similarly evident. $\sigma_1 | \ldots | \sigma_n$ is even-branching because if $sa, sb \in \sigma_1 | \cdots | \sigma_n$, and $ta, t'b$ are interaction sequences such that $ta{\restriction}A_i \to A_{i+1} \in \sigma_i, t'b{\restriction}A_i \to A_{i+1} \in \sigma_i$ for each i and $s = t{\restriction}A_1 \to A_{n+1} = t'{\restriction}A_1 \to A_{n+1}$ then $t = t'$ and hence $a = b$. □

Composition of morphisms in our category of games is the binary composition of strategies: to show that this is associative, we establish that $(\sigma_1 | \sigma_2) | \sigma_3 = \sigma_1 | \sigma_2 | \sigma_3 = \sigma_1 | (\sigma_2 | \sigma_3)$. Key to this is showing that if t is an interaction sequence on $A_1, \ldots, A_{n+1}$, then $t{\restriction}A_1 \to A_{n+1}$ is a legal sequence if and only if t is itself a legal sequence. First, note that it is evident that t satisfies the downwards closure condition if and only if $t{\restriction}A_i$ is down-closed for each i, if and only if $t{\restriction}A_1 \to A_{n+1}$ is down-closed. In the following lemmas, we will not assume down-closure.

Say that an interaction sequence t on $A_1, \ldots, A_{n+1}$ satisfies the *switching condition* if for every even-prefix $s \sqsubseteq t$, $s{\restriction}A_i \to A_{i+1}$ is even-length for each i, and for every odd-length prefix $s \sqsubseteq t$, $s{\restriction}A_1 \to A_{n+1}$ is of odd length. (In other words, if a Player move in t is in $A_i \to A_{i+1}$ then the preceding move is in $A_i \to A_{i+1}$ and if an Opponent move in t is in $A_1 \to A_{n+1}$ then the preceding move (if any) is in $A_1 \to A_{n+1}$.)

LEMMA 4.5. *If s is an interaction sequence on $A_1, \ldots, A_{n+1}$ and $s{\restriction}A_1 \to A_{n+1}$ is alternating, then (i) s is alternating and (ii) s satisfies the switching condition.*

PROOF. By induction on length (with trivial base cases of empty or singleton sequences):

—Given sab of odd length, by hypothesis, a is a Player move, and $sa \upharpoonright A_i \to A_{i+1}$ is even-length for each i, and so by alternation, b must be an Opponent move in some i, as required. If b is a move in A_i for some $1 < i \leq n$, then it is a copy of a and so by hypothesis on s, $sab \upharpoonright A_1 \to A_{n+1}$ is odd-length. Otherwise b is a move in $A_1 \to A_{n+1}$ and so $sab \upharpoonright (A_1 \to A_{n+1})$ is odd-length by assumption that it is alternating.
—Given sab of even length, by hypothesis a is an Opponent move in $A_i \to A_{i+1}$ for some i, and $sa \upharpoonright (A_1 \to A_{n+1})$ is odd-length. Then b cannot be an Opponent move in A_i for some $1 < i \leq n$, as this contradicts the copycat condition, and it cannot be an Opponent move in $A_1 \to A_{n+1}$ as this contradicts the hypothesis that $sa \upharpoonright A_1 \to A_{n+1}$ is odd-length and alternating. Hence, b is a Player move: by the alternation condition it must also be a move in $A_i \to A_{i+1}$ and so $sa \upharpoonright A_j \to A_{j+1}$ is alternating for each j as required. □

Proof of the following converse is similar:

LEMMA 4.6. *If s is an alternating interaction sequence on $A_1, \ldots, A_{n+1}$, then it satisfies the switching condition, and $s \upharpoonright A_1 \to A_j$ is alternating for any $j > 1$.*

We now prove similar results with respect to the well-bracketing condition.

LEMMA 4.7. *For any alternating interaction sequence s on $A_1, \ldots, A_{n+1}$, if $s \upharpoonright A_1 \to A_{n+1}$ is well bracketed, then s is well bracketed.*

PROOF. By induction on the length of s: suppose $s = s'qta$, where t is complete, we need to show that $q \rhd a$.

—Suppose a is a Player move in $A_i \to A_{i+1}$, then we know by the switching condition that the preceding move is in $A_i \to A_{i+1}$. If this is a question (i.e., t is empty), then by well-bracketedness of $s \upharpoonright A_i \to A_{i+1}$, $q \rhd a$ as required. If it is an answer—that is, $t = t'q'ra'$, where r is complete—then by hypothesis $sqt'a$ is well-bracketed and hence $q \rhd a$ as required.
—Suppose a is an Opponent move in A_i for some $1 < i \leq n$. Then, it is an answer move in A_i (all Opponent hole moves become questions under quantification) and a copy of the previous Player move a', which must also be an answer. By hypothesis, a' answers a question q' such that $q' \rhd a'$, and by the copycat condition, q' must be a copy of the preceding move, which must be q, and so $q \rhd a$ as required. If a is an answer move in $A_1 \to A_{n+1}$ then by the switching condition, the preceding Player move is in $A_1 \to A_{n+1}$. If this is an answer (i.e., t is empty), then by well-bracketedness of $s \upharpoonright A_1 \to A_{n+1}$, $q \rhd a$ as required. If it is an answer—that is, $t = t'q'ra'$, where r is complete—then by hypothesis $sqt'a$ is well bracketed and hence $q \rhd a$ as required. □

Proof of the following lemma is similar:

LEMMA 4.8. *For any alternating interaction sequence s on $A_1, \ldots, A_{n+1}$, if s is well bracketed, then $s \upharpoonright (A_1 \to A_i)$ is well bracketed for each $i > 1$.*

LEMMA 4.9. *Suppose r is a legal interaction sequence on A_1, A_2, A_3, and s is a legal interaction sequence on A_1, A_3, A_4 such that $r \upharpoonright A_1 \to A_3 = s \upharpoonright A_1 \to A_3$. Then there is a legal interaction sequence $r \leftslash s$ on A_1, A_2, A_3, A_4 such that $(r \leftslash s) \upharpoonright (A_1 \to A_2) \uplus (A_2 \to A_3) = r$ and $(r \leftslash s) \upharpoonright (A_1 \to A_3) \uplus (A_3 \to A_4) = s$.*

PROOF. By induction on the combined length of r and s.

(1) If $r = r'a$, where a is a move in A_2, then define $r \leftslash s = (r' \leftslash s)a$.
(2) If $s = s'a$, where a is a move in A_4, then define $r \leftslash s = (r \leftslash s')a$.
(3) Otherwise, $r = r'a$ and $s = s'a$ for some common move a: define $r \leftslash s = (r' \leftslash s')a$.

Note that by the switching condition, conditions 1 and 2 are mutually exclusive, and that $r \leftthreetimes s$ is an Opponent (A_2^-, A_2^+) and (A_3^-, A_3^+) copycat. □

LEMMA 4.10. $\sigma_1|\sigma_2|\sigma_3 = (\sigma_1|\sigma_2)|\sigma_3$.

PROOF. If $s \in \sigma_1|\sigma_2|\sigma_3$, then there exists an interaction sequence r on A_1, A_2, A_3, A_4 such that $r\restriction A_i \to A_{i+1} \in \sigma_i$ for $i \in \{1, 2, 3\}$ and $r\restriction A_1 \to A_4 = s$. By Lemma 4.8, $r\restriction A_1 \to A_3$ is a legal sequence, and therefore in $\sigma_1|\sigma_2$. Hence, $t = r\restriction (A_1 \to A_3) \uplus (A_3 \to A_4)$ is a legal interaction sequence on A_1, A_3, A_4 such that $t\restriction A_1 \to A_3 \in \sigma_1|\sigma_2$ and $t\restriction A_3 \to A_4 \in \sigma_3$. Thus, $s = t\restriction A_1 \to A_4$ is in $(\sigma_1|\sigma_2)|\sigma_3$ as required.

Suppose $s \in (\sigma_1|\sigma_2)|\sigma_3$. Then, there exists a legal interaction sequence r on A_1, A_3, A_4 such that $r\restriction A_1 \to A_3 \in (\sigma_1|\sigma_2)$, $r\restriction A_3 \to A_4 \in \sigma_3$ and $r\restriction A_1 \to A_4 = s$. Hence, there exists a legal interaction sequence t on A_1, A_2, A_3 such that $t\restriction A_1 \to A_2 \in \sigma_1$ and $t\restriction A_2 \to A_3 \in \sigma_2$ and $t\restriction A_1 \to A_3 = r\restriction A_1 \to A_3$. Then, $r \leftthreetimes t$ is an interaction sequence on A_1, A_2, A_3, A_4 such that $(r \leftthreetimes t)\restriction A_i \to A_{i+1} \in \sigma_i$ for $i \in \{1, 2, 3\}$ and so $s = (r \leftthreetimes t)\restriction A_1 \to A_4 \in \sigma_1|\sigma_2|\sigma_3$ as required. □

By symmetry, $\sigma_1|\sigma_2|\sigma_3 = \sigma_1|(\sigma_2|\sigma_3)$. Thus, composition in our category of games is associative. It is straightforward to check that the identity strategy is an identity for composition.

4.1. Reindexing, Universal Quantification and Instantiation

Given $f : \mathbb{N} \to \mathbb{N}$, the (identity-on-morphisms) f-reindexing functor sends the game A to $(M_A, f_{QA} \circ \lambda_A, \rhd_A)$. (Reindexing defines a functor from the endofunctions on $\mathbb{N}$ to the endofunctors on $\mathcal{G}$, monoidal with respect to composition.) Let $\mathcal{G}_0$ be the full subcategory of $\mathcal{G}$ consisting of 0-closed games: the successor-reindexing and predecessor-reindexing functors correspond to an isomorphism $S : \mathcal{G} \cong \mathcal{G}_0 : P$. $\mathcal{G}_0$ is also a *reflective subcategory* of $\mathcal{G}$.

PROPOSITION 4.11. *The inclusion $J : \mathcal{G}_0 \to \mathcal{G}$ has a left adjoint $\forall_0 : \mathcal{G} \to \mathcal{G}_0$.*

PROOF. For any 0-closed game A, and game B, $\forall_0(A \to B) = A \to \forall_0 B$. Hence, there is a natural isomorphism between $\mathcal{G}(J(A), B)$ and $\mathcal{G}_0(A, \forall_0(B))$. □

The instantiation of types for universally quantified type variables in *terms* will be interpreted by extending strategies to the instantiated games by playing *copycat* between the games substituted for a question and its closing answer move. (See Figure 2.)

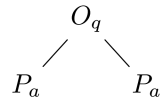
Formally, given a strategy $\sigma : A \to B$, define $\sigma[C] : A[C] \to B[C]$ to be the set of even-length legal sequences s on $A[C] \to B[C]$ such that:

—$s\restriction A \to B \in \sigma$. (i.e., $p^*(s) \in \sigma$, where $p(m \cdot 0^*) = m$ if m is a move of $A \to B$, and p is undefined otherwise).
—For any 0-hole moves u, v of $A \to B$ such that $(u, v) \in \mathsf{Cl}_{A \to B}(s\restriction A \to B)$, s is a Player (u, v)-copycat.

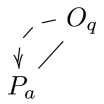
This is a well-defined strategy—it is evidently even-prefix closed and satisfies the determinacy condition since the response to any move of $A \to B$ (or instantiated initial move of C) is determined by σ, and the response to any non-initial instantiated move of C is determined by the copycat condition.

To show that instantiation is compositional, we use the following observation: Suppose t is a well-bracketed interaction sequence on A, B, D, and $(u, v) \in \mathsf{Cl}_{A \to D}(t\restriction A \to D)$. Then, by the definition of interaction sequences and the closure relation there exists a (unique) chain of moves $w_0, \ldots, w_{2n+1}$ such that $w_0 = u$ and $w_{2n+1} = v$, and $(w_{2i}, w_{2i+1}) \in \mathsf{Cl}_{A \to B \uplus B \to D}(s)$, which we may call a *bridging sequence* for (u, v) in t. Note that if u, v are i-hole moves, then all intermediate moves in the bridging sequence are i-hole moves in B, and that for $i > 0$, w_{2i} is a copy by Opponent of w_{2i-1}.

Instantiating the game $\mathbb{B}$:



into the identity strategy on $\bullet_0 \to \bullet_0$:



gives the identity on $\mathbb{B} \to \mathbb{B}$:

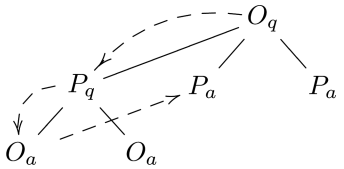


Fig. 2. Instantiation to strategies: An example.

LEMMA 4.12. *Given* $\sigma : A \to B$ *and* $\tau : B \to C$, $\sigma[C]|\tau[C] = (\sigma|\tau)[C]$

PROOF. Suppose $s \in \sigma[C]|\tau[C]$. Then, there exists an interaction sequence t on $A[C], B[C], D[C]$ such that $t \upharpoonright A[C] \to B[C] \in \sigma[C]$ and $t \upharpoonright B[C] \to D[C] \in \tau[C]$. Hence, $t \upharpoonright A \to D = (t \upharpoonright ((A \to B) \uplus (B \to D))) \upharpoonright A \to D \in \sigma|\tau$. We need to show that, if $(u, v) \in \mathsf{Cl}_{A\to D}(t \upharpoonright A \to D)$, then t is a Player (u, v)-copycat. Consider the bridging sequence $w_0, \ldots, w_{2n+1}$ for u, v in t. For each $i \leq n$, t is a Player (u_{2i}, u_{2i+1})-copycat because $t \upharpoonright A[C] \to B[C] \in \sigma[C]$ and $t \upharpoonright B[C] \to D[C] \in \tau[C]$. For each $i < n$, t is an Opponent (u_{2i+1}, u_{2i+2})-copycat, because t is an interaction sequence. Thus, t is a Player (u, v)-copycat as required.

Suppose $s \in (\sigma|\tau)[C]$. Then, there exists an interaction sequence t on A, B, D such that $s \upharpoonright A \to D = t \upharpoonright A \to D$. We show by induction on the length of s that there exists an interaction sequence $t[s] \in \sigma[C]|\tau[C]$ such that $t[s] \upharpoonright A[C] \to D[C] = s$.

—If $s = s'ab$, where a, b are not (noninitial) 0-hole moves of $A \to D$, then $t = t'at''b$, where t'' consists of interaction in B and thus we may define $t[s] = t'[s']at''b$.
—If $s = s'(u \cdot w')(v \cdot w')$, where $(u, v) \in \mathsf{Cl}_{A\to D}(s) \cap H_{A\to D}(0)$, then there is a bridging sequence $w_0, \ldots, w_{2n+1}$ for u, v in t, and thus we may define $t[s] = t[s'](w_0 \cdot w') \cdots (w_{2n+1} \cdot w')$. □

LEMMA 4.13. *For any game* A, $\mathsf{id}_A[C] = \mathsf{id}_{A[C]}$.

PROOF. We prove by induction that $s \in \mathsf{id}_A[C]$ if and only if $s \in \mathsf{id}_{A[C]}$. □

Hence, we have shown that for any well-opened context game C the instantiation functor $\_[C] : \mathcal{G} \to \mathcal{G}$ sending A to $A[C]$ and $\sigma : A \to B$ to $\sigma[C] : A[C] \to B[C]$ is well defined. We shall also make use of the following facts about the instantiation functor, which follow directly from the definitions above:

LEMMA 4.14. *The instantiation* $[\bullet_0]$ *is the identity functor on* $\mathcal{G}$.

LEMMA 4.15. *For any well-opened games* B, C, *the composition* $\_[C] \circ \_[B] = \_[B[C]]$.

4.2. Further Categorical Structure

We define *symmetric monoidal* structure on $\mathcal{G}$: the tensor product $A \otimes B$ is the disjoint union $01.A \cup 10.B$. Given $\sigma : A \to C$ and $\tau : B \to D$, we define $\sigma \otimes \tau : A \otimes B \to C \otimes D =$

$$\{s \in L_{A\otimes B \to C \otimes D} \mid s\restriction A \to C \in \sigma \wedge s\restriction B \to D \in \tau\}.$$

The unit I for the tensor is the "empty game"—that is, the unique game over an empty set of moves (note that this is a terminal object). There are copycat strategies yielding the required isomorphisms making $(\mathcal{G}, \otimes, I)$ a symmetric monoidal category. (Proof that $\otimes$ is bifunctorial and satisfies the relevant coherence diagrams follows, for example, McCusker [1996].) Each instantiation functor is strict monoidal. In the following, we shall elide associativity and unit isomorphisms, as if in a strict SMC.

Given games A and well-opened B, define the game $A \multimap B = 1.A \cup 00.B$. This grafts (a disjoint copy of) A onto the root of B—if $o \in \{0\}^*$ is the $\preceq$-least element of B, then $00.o$ is the least element of $A \multimap B$ and all other elements $1.a$ and $00.b$ are incomparable. Opponent moves (even-length sequences) in A become Player moves in $A \multimap B$ and vice-versa.

LEMMA 4.16. *The well-opened negative games form an* exponential ideal *in* $(\mathcal{G}, \otimes, I)$.

PROOF. If B is well-opened, then for any A, $A \multimap B$ is well-opened, and the exponential of B by A: the evident correspondence between legal sequences on $C \to (A \multimap B)$ and on $C \otimes A \to B$ yields the required natural isomorphisms. □

$\mathcal{G}$ is also cpo-enriched with the inclusion order on strategies: let $\bot_{A,B}$ be the least element of $\mathcal{G}(A, B)$—the strategy containing only the empty sequence.

As an example, we may observe how the *lifted sum* of games arises naturally in our model, corresponding to the encoding of sums in second-order type theory. Let $\mathcal{G}_t$ be the subcategory of $\mathcal{G}$ consisting of well-opened context games and total morphisms. (A morphism $f : A \to B$ of a cpo-enriched category is *total* if for any $g : B \to C$, $f;g = \bot_{A,C}$ implies $g = \bot_{B,C}$: concretely f is total if Player always responds to the unique initial move in B with the unique initial move in A.) For a family of games $\{A_i \mid i < n\}$, let $\Sigma_{i<n}A_i = P(\forall_0(((S(A_0) \multimap \bullet_0) \multimap \cdots \multimap (S(A_{n-1}) \multimap \bullet_0)) \multimap \bullet_0))$ (where P and S are the predecessor and successor reindexings). This construction is familiar from McCusker [1996] as the *lifted sum* in which Opponent asks an initial question, then Player answers with a choice of $i < n$ and play continues in A_i. It is a weak coproduct on $\mathcal{G}$—$\Sigma_\_$ is left adjoint to the inclusion of the category of very-well-opened games and total strategies in $\mathsf{Fam}(\mathcal{G})$ (the completion of $\mathcal{G}$ with finite coproducts). (A well-opened game is very-well-opened if the initial move enables only Player moves.)

5. THE CO-FREE COMMUTATIVE COMONOID

In order to extend our semantics to intuitionistic function types and mutable variables, we introduce a "bang" construction on games, à la linear logic.

Definition 5.1. Given a well-opened game A let $!A$ be the well-opened game $(11)^*.00.A$—that is, $00.A \cup 1100.A \cup 111100.A \cup \cdots$.

In other words $!A$ consists of countably many distinctly tagged copies of A: the initial moves in each copy (of the form $1^{2n} \cdot 0^{2m}$) must be played (by Opponent) in order of the size of n.

We may identify the following strategies:

—$\eta_A : !A \to I$—the empty strategy (terminal map),
—$\delta_A : !A \to !A \otimes !A$—which plays copycat between $!A \otimes !A$ and $!A$, opening a fresh copy of A in $!A$ for each thread opened on either side of $!A \otimes !A$.

These correspond to concrete interpretations of the weakening and contraction rules of linear logic and make $(!A, \delta_A, \eta_A)$ a *commutative comonoid*—indeed we may identify:

—a morphism $\epsilon_A : !A \to A$ (which opens one thread of $!A$ and plays copycat with A) and
—for any commutative comonoid (B, δ_B, η_B), and $\sigma : B \to A$, a strategy $\sigma^\dagger : B \to !A$ which plays as σ in each copy of A.

corresponding to the dereliction and promotion rules and making $(!A, \delta_A, \eta_A)$ the *cofree commutative comonoid* on A (see here). Since the bang is also key to our interpretation of higher-order state, rather than formalising the definitions of this structure combinatorically, we shall make use of constructions described in Laird [2002], based on the fact that it is a *minimal invariant* for an action $\_ \oslash \_ : \mathcal{G}_t \times \mathcal{G} \to \mathcal{G}_t$ of the monoidal category $\mathcal{G}$ on its subcategory of total morphisms.

First, we note that for any game A, the exponential restricts to a functor $A \multimap \_ : \mathcal{G}_t \to \mathcal{G}_t$ with the following property.

LEMMA 5.2. *For any A, the functor $A \multimap \_ : \mathcal{G}_t \to \mathcal{G}_t$ has right and left adjoints, which commute with $A \multimap \_$ (and therefore coincide)—that is, $A \multimap \_$ has a dual in the monoidal category of endofunctors on $\mathcal{G}_t$.*

PROOF. For any well-opened A, define $A \oslash B = 00.A + 11.B$ (i.e. equivalent as a partial order to $B \multimap A$, but without swapping Player/Opponent polarity in B). Then, there are evident adjunctions:

$$\frac{\mathcal{G}_t(A \oslash B, C)}{\mathcal{G}_t(A, B \multimap C)} \qquad \frac{\mathcal{G}_t(B \multimap A, C)}{\mathcal{G}_t(A, C \oslash B)}$$

and a natural isomorphism $\gamma : B \multimap (A \oslash B') \cong (B \multimap A) \oslash B'$. □

In the terminology of Laird [2002], this commuting adjunction defines a *sequoid*, i.e.:

—$\_ \oslash \_ : \mathcal{G}_t \times \mathcal{G} \to \mathcal{G}_t$ is an *action* of the SMC $(\mathcal{G}, \otimes, I)$ on $\mathcal{G}_t$.
—There is a natural transformation $\nu : J\_ \otimes J\_ \to J(\_ \oslash J\_)$, where $J : \mathcal{G}_t \to \mathcal{G}$ is the inclusion functor: $\nu_{A,B} = \Lambda^{-1}(J(\eta_{A,B}))$, where $\eta_{A,B} : A \to B \multimap (A \oslash B)$ is the unit of the relevant adjunction and $\Lambda^{-1} : \mathcal{G}(A, B \multimap (A \oslash B)) \to \mathcal{G}(A \otimes B, A \oslash B)$ is uncurrying of the exponential.

This sequoid has the further property of "sequential decomposability", which we shall use to define a comonoid.

LEMMA 5.3. *For any well-opened games A, B, $A \otimes B$ is a Cartesian product of $A \oslash B$ and $B \oslash A$, with projections $\nu_{A,B} : A \otimes B \to A \oslash B$ and $\theta_{A,B}; \nu_{B,A} : A \otimes B \to B \oslash A$ (where $\theta_{A,B} : A \otimes B \to B \otimes A$ is the symmetry isomorphism of $\otimes$).*

So, in particular, there is a diagonal morphism $\Delta_B : B \oslash B \to B \otimes B = \langle \mathsf{id}_{B \oslash B}, \mathsf{id}_{B \oslash B} \rangle$ for any well-opened game B.

A *minimal invariant* for an endofunctor of cpo-enriched categories $F : \mathcal{C} \to \mathcal{C}$ is an object B with an isomorphism $\mathsf{in} : F(B) \cong B : \mathsf{out}$ such that the least fixedpoint for the operation taking $f : B \to B$ to $\mathsf{out}; F(f); \mathsf{in}$ is the identity on B.

LEMMA 5.4. *For any well-opened A, $!A$ is a minimal invariant for the functor $J(A \oslash \_) : \mathcal{G} \to \mathcal{G}$.*

PROOF. $!A = (11)^*.00.A = 00.A \cup 11.(11)^*.00.A = A \oslash !A$—minimal invariance follows by continuity. □

We may now use minimal invariance and sequoidal structure to define the comonoid structure described earlier.

Definition 5.5. For any well-opened A, the comonoid $(!A, \delta_A : !A \to !A \otimes !A, \eta_A : !A \to I)$ is given as follows:

—$\delta_A : !A \to !A \otimes !A$ is the least fixed point of the endofunction on $\mathcal{G}(!A, !A \otimes !A)$ sending f to:

$$!A \cong A \oslash !A \xrightarrow{\mathsf{id}_A \oslash f} A \oslash (!A \otimes !A) \cong (A \oslash !A) \oslash !A \cong !A \oslash !A \xrightarrow{\Delta_{!A}} !A \otimes !A$$

—$\eta_A : !A \to I$ is the terminal map t_A.

For a symmetric monoidal category $(\mathcal{C}, \otimes, I)$, let $\mathsf{Comon}(\mathcal{C})$ be the category of commutative comonoids and comonoid morphisms of $\mathcal{C}$.

PROPOSITION 5.6. *For any well-opened A, $!A$ is the cofree commutative comonoid on A.*

PROOF. In other words, if $U : \mathsf{Comon}(\mathcal{G}) \to \mathcal{G}$ is the forgetful functor, then for any commutative comonoid (B, δ_B, η_B) there is an isomorphism (natural in B) between $\mathcal{G}(U(B), A)$ and $\mathsf{Comon}(\mathcal{G})(B, !A)$ given by a morphism $\epsilon_A : U(!A) \to A$ and an operation sending each morphism $f : U(B) \to A$ in $\mathcal{G}$ to $f^\dagger : B \to !A$ such that $f^\dagger ; \epsilon_A = f$ and $\epsilon_A^\dagger = \mathsf{id}_{!A}$.

ϵ_A is defined by:

$$!A \cong A \oslash !A \xrightarrow{\mathsf{id}_A \oslash \eta_A} A \oslash I \cong A$$

and given $f : U(B) \to A$, define $f^\dagger : B \to !A$ to be the least fixed point of the operation sending $g : B \to !A$ to:

$$B \xrightarrow{\delta_B} B \otimes B \xrightarrow{f \otimes g} A \otimes !A \xrightarrow{\nu_{A,!A}} A \oslash !A \cong !A$$

These identities follow by straightforward induction. □

COROLLARY 5.7. *The (free commutative comonoids on) well-opened games form an exponential ideal in the Cartesian category* $\mathsf{Comon}(\mathcal{G})$.

PROOF. We have $\mathsf{Comon}(\mathcal{G})(A \otimes B, !C) \cong \mathcal{G}(U(A) \otimes U(B), C) \cong \mathcal{G}(U(A), U(B) \multimap C) \cong \mathsf{Comon}(\mathcal{G})(A, !(U(B) \multimap C))$. □

Finally, note that the instantiation functor strictly preserves this structure—that is,

—$\_[B]$ lifts to an endofunctor on $\mathsf{Comon}(\mathcal{G})$ (sending (A, δ, η) to $(A[B], \delta[B], \eta[B])$) which preserves products and well-opened exponentials.

5.1. Semantics of References

Variables of System $\mathrm{F}_{\mathtt{ref}}$ denote mutable reference cells. Following Abramsky et al. [1998], our semantics is based on representing such a cell as a strategy on the game $\S A = !(!A \multimap \Sigma) \otimes !A$ (which is a Cartesian product in our category of comonoids), so that writing to the cell corresponds to applying the left projection from this product, reading from it corresponds to right-projection, and declaration of a variable (composition with the cell strategy in $\mathcal{G}$) connects writes to reads appropriately. Thus, we need to define a family of strategies $\mathsf{cell} : !A \to \S A$, uniform in A. Informally, these are a simplified version of the cell strategy defined in Abramsky et al. [1998]: if Opponent requests a "write" by asking the initial question on the left, Player responds with the unique answer to this question "Ok" and if Opponent requests a read by opening a thread of $!A$ on the right, then Player responds by either:

—Playing copycat with the last-opened write thread, if this exists, or,
—Playing copycat with the source copy of $!A$, otherwise.

The polymorphic version $\mathsf{cell}_{\bullet_0}$ encapsulates this copycat behaviour in the following play:

$$\begin{array}{ccccccc} !\bullet_0 & \to & (!\bullet_0 & \multimap & \Sigma) & \otimes & !\bullet_0 \\ & & & & & & O_q \\ P_a & & & & & & \\ & & & & O_q & & \\ & & & & P_a & & \\ & & & & & & O_q \\ & & P_a & & & & \end{array}$$

To define this strategy formally, we shall give a construction of it based on Laird [2002], using the properties of the sequoid, and the ! as a minimal invariant.

Recall that $\_ \oslash A : \mathcal{G}_t \to \mathcal{G}_t$ is left (as well as right) adjoint to $A \multimap \_$: the counit $\eta_\Sigma : \Sigma \to (A \multimap \Sigma) \oslash A$ of this adjunction is the basis of the cell strategy.

Definition 5.8. Let $\mathsf{cell}_A : !A \to !(!A \multimap \Sigma) \otimes !A$ be the least fixed point of the map $\Phi : \mathcal{G}(!A, !(!A \multimap \Sigma) \otimes !A) \to \mathcal{G}(!A, !(!A \multimap \Sigma) \otimes !A)$ sending f to the pairing of:

$$\begin{aligned} !A &\cong A \oslash !A \xrightarrow{A \oslash f} A \oslash (!(A \multimap \Sigma) \otimes !A) \cong A \oslash (!A \otimes !(!A \multimap \Sigma)) \cong (A \oslash !A) \oslash !(!A \multimap \Sigma) \\ &\cong !A \oslash !(!A \multimap \Sigma) \end{aligned}$$

(reading) and

$$!A \xrightarrow{t_{!A}} \mathbf{1} \xrightarrow{\top} \Sigma \xrightarrow{\eta_{\Sigma,!A}} (!A \multimap \Sigma) \oslash !A \xrightarrow{(!A \multimap \Sigma) \oslash f} (!A \multimap \Sigma) \oslash (!(!A \multimap \Sigma) \otimes !A) \cong !(!A \multimap \Sigma) \oslash !A$$

(writing), where $\top : \mathbf{1} \to \Sigma$ is the universal quantification of the curried identity strategy—recall that $!(!A \multimap \Sigma) \otimes !A$ is the Cartesian product of $!(!A \multimap \Sigma) \oslash !A$ and $!A \oslash !(!A \multimap \Sigma)$ in $\mathcal{G}$.

Thus, we have a family of cell strategies for each context game, which is invariant under instantiation—that is, $\mathsf{cell}_A[B] = \mathsf{cell}_{A[B]}$ for each A and well-opened B—since instantiation preserves sequoidal structure. Each cell strategy satisfies key categorical equations defining the behaviour of a reference cell: We define assignment and dereferencing strategies which take left/right projections from the top copy of the cell, and behave as the identity on the remainder: Let $\mathsf{read}_A : \S A \to A \oslash \S A = (\delta_{\S A}; \nu_{\S A, \S A}); (\pi_r; \epsilon_A \oslash \S A)$ and $\mathsf{write} : \S A \to (!A \multimap \Sigma) \oslash \S A = (\delta_{\S A}; \nu_{\S A, \S A}); (\pi_l; \epsilon_A \oslash \S A)$.

Reading. $\mathsf{cell}_A; \mathsf{read}_A : !A \to A \oslash \S A = (\delta_{!A}; \nu_{!A,!A}); (\epsilon_A \oslash \mathsf{cell}_A)$ (i.e., reading from the cell returns the value stored there and leaves the cell unchanged).

Writing. $\mathsf{cell}_A; \mathsf{write}_A : !A \to (!A \multimap \Sigma) \oslash \S A = t_{!A}; \Lambda((\top \otimes \mathsf{cell}_A); \nu_{\Sigma, \S A}); \gamma_{!A, \Sigma, \S A}$—that is, selecting the write component returns a method which updates the cell and returns $\top$.

6. DENOTATIONAL SEMANTICS OF SYSTEM $\mathrm{F}_{\mathrm{REF}}$

We may now define the semantics of System $\mathrm{F}_{\mathtt{ref}}$: we first give a direct interpretation, and will then observe that it is an instance of a more general hyperdoctrine construction. Types $X_0, \ldots, X_{n-1} \vdash T$ may be interpreted as objects of $\mathsf{Comon}(\mathcal{G})$, as follows:

—$[\![X_0, \ldots, X_{n-1} \vdash X_0]\!] = \bullet_0$.
—$[\![X_{\pi(0)}, \ldots, X_{\pi(n-1)} \vdash T]\!] = \Phi(\pi)([\![X_0, \ldots, X_{n-1} \vdash T]\!])$ (where $\Phi(\pi)$ is the π-reindexing functor).
—$[\![\Theta \vdash \mathsf{com}]\!] = \Sigma$.

—$[\![\Theta \vdash S \to T]\!] = !{[\![\Theta \vdash S]\!]} \multimap [\![\Theta \vdash T]\!]$.
—$[\![\Theta \vdash \forall X.T]\!] = P(\forall_0([\![X, \Theta \vdash T]\!]))$—universal quantification followed by predecessor-reindexing.

Terms $\Theta; x_1 : S_1, \ldots, x_n : S_n \vdash M : T$ are interpreted as morphisms from $\S[\![\Theta \vdash S_1]\!] \otimes \cdots \otimes \S[\![\Theta \vdash S_n]\!]$ to $![\![\Theta \vdash T]\!]$ in $\mathsf{Comon}(\mathcal{G})$, which we may identify with the associated morphism from $\S[\![\Theta \vdash S_1]\!] \otimes \cdots \otimes \S[\![\Theta \vdash S_n]\!]$ to $[\![\Theta \vdash T]\!]$ in $\mathcal{G}$, defined as follows:

—$[\![\Theta; \Gamma \vdash () : \mathsf{com}]\!] = \top : [\![\Gamma]\!] \to \Sigma$—the $\top$ strategy on Σ.
—$[\![\Theta; \_ \vdash \Omega : \forall X.X]\!] = \bot$—the empty strategy.
—$[\![\Theta; \Gamma, x : T \vdash x : T]\!] : [\![\Gamma]\!] \otimes \S[\![T]\!] \to [\![T]\!] = \pi_r; \pi_r; \epsilon$—right projection from $\S[T]\!]$.
—$[\![\Theta; \Gamma, x : T \vdash \mathtt{set}(x) : T \to \mathsf{com}]\!] : [\![\Gamma]\!] \otimes \S[\![T]\!] \to [\![T \to \mathsf{com}]\!] = \pi_r; \pi_l; \epsilon$—left projection from $\S[\![T]\!]$.
—$[\![\Theta; \Gamma \vdash \Lambda X.M : \forall X.T]\!] : [\![\Gamma]\!] \to [\![\Theta \vdash \forall X.T]\!] = \forall_n [\![\Theta, X; \Gamma \vdash M]\!]$—the action of the universal adjunction.
—$[\![\Theta; \Gamma \vdash M\{S\} : T[S/X]]\!] : [\![\Gamma]\!][[\![\Theta \vdash S]\!]] \to [\![T]\!][[\![\Theta \vdash S]\!]] = [\![\Theta; \Gamma \vdash M]\!][[\![\Theta \vdash S]\!]]$—the instantiation functor.
—$[\![\Theta; \Gamma \vdash M; N : T]\!] : [\![\Gamma]\!] \to [\![T]\!] = \langle [\![\Theta; \Gamma \vdash M : \mathsf{com}]\!]; \mathsf{seq}_T, [\![\Theta; \Gamma \vdash N : T]\!] \rangle; \mathsf{app}$, where $\mathsf{seq}_T : \Sigma \to [\![T \to T]\!]$ is the sequential composition strategy.
—$[\![\Theta; \Gamma \vdash \lambda x^S.M : S \to T]\!] : [\![\Gamma]\!] \to [\![S \to T]\!] = \Lambda(([\![\Gamma]\!] \otimes \mathsf{cell}_{[\![S]\!]}); [\![\Theta; \Gamma, x : S \vdash M : T]\!])$—composition (in $\mathcal{G}$) with the strategy cell, followed by currying.
—$[\![\Theta; \Gamma \vdash M\,N : T]\!] : [\![\Gamma]\!] \to [\![T]\!] = \langle [\![\Theta; \Gamma \vdash M : S \to T]\!], [\![\Theta; \Gamma \vdash N : S]\!] \rangle; \mathsf{app}$—the standard interpretation of application for an exponential object.

6.1. Soundness

To prove soundness of our semantics, we shall first show that it may viewed as (essentially) an instance of the notion of a categorical model of System F typing introduced by Seely [1987] and simplified by Pitts [1988]. First, we generalize instantiation to multiple games. Define the n-fold instantiation functor $[B_0, \ldots, B_{n-1}]$ (which substitutes each B_i into the i-holes of its argument) by induction on n as follows:

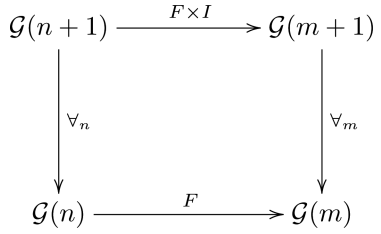
—$A[\_] = A$
—$A[B_0, \ldots, B_{n-1}] = P(A[S^n(B_0)])[B_1, \ldots, B_{n-1}]$

Let $\mathcal{I}$ be the category in which objects are natural numbers, and morphisms from m to n are m-tuples of n-context games. The identity on n is the tuple $[\bullet_0, \ldots, \bullet_{n-1}]$ and composition of $(B_0, \ldots, B_m)$ with $(A_0, \ldots, A_l)$ is $(A_0[B_0, \ldots, B_m], \ldots A_l[B_0, \ldots, B_m])$. That this is a category follows from Lemmas 4.14 and 4.15. It has finite products (arithmetic sums) and is finitely generated by them from the object 1. We define an $\mathcal{I}$-indexed category with (specified) finite products: the identity-on-morphisms functor $(\_)^*$ from $\mathcal{I}^{OP}$ into the category of categories with finite products sending n to the subcategory $\mathcal{G}(n)$ of $\mathsf{Comon}(\mathcal{G})$ consisting of (comonoids on) n-context games, and $(B_0, \ldots, B_m)$ to the instantiation functor $\_[B_0, \ldots, B_m]$.

PROPOSITION 6.1. *The inclusion functors $J_n : \mathcal{G}(n) \to \mathcal{G}(n+1)$ have an indexed left adjoint $\forall_n : \mathcal{G}(n+1) \to \mathcal{G}(n)$.*

PROOF. For any n-context game A, and $n+1$-context game B, $\forall_n(J_n(A) \to B) = A \to \forall_n B$. Hence, there is a a natural isomorphism between $\mathcal{G}(n+1)(J_n(A), B)$ and $\mathcal{G}(n)(A, \forall_n(B))$.

Moreover, these satisfy the *Beck-Chevalley* condition and so form an indexed adjunction. In other words, for each instantiation functor $F : \mathcal{G}(n) \to \mathcal{G}(m)$, the following square commutes:



If F is the instantiation $[B_0, \ldots, B_{n-1}]$, this boils down to the requirement that $\forall_m A[B_0, \ldots, B_{n-1}, \bullet_m] = (\forall_n A)[B_0, \ldots, B_{n-1}]$, which follows from Lemma 4.14. □

This completes the characterization of our semantics as a Seely-Pitts hyperdoctrine model.

6.2. Soundness for System $\mathbf{F}_{\mathtt{ref}}$

Soundness of β-reduction for type instantiation follows from the categorical construction above. To prove soundness of the declaration, assignment and dereferencing rules, we interpret configurations by defining a *partial trace operator* [Malherbe et al. 2011] on our symmetric monoidal category of games using the sequoidal structure that has already been identified. Given a morphism $f : A \otimes B \to C \otimes B$, where C is well opened, define

$$\mathsf{trace}(f) : A \to C = \Lambda(f; \nu_{C,B}); \epsilon_{B,C}$$

where $\epsilon_{B,C} : (B \multimap (C \oslash B)) \to C$ is the co-unit of the adjunction $\_ \oslash B \dashv B \multimap \_$. Then, given a configuration $(M|((a_1, N_1), \ldots (a_k, N_k)))$ where $M, N_1 : T_1, \ldots, N_k : T_k$ may contain the names $a_1, \ldots, a_k$, we define $[\![M|((a_1, N_1), \ldots (a_k, N_k))]\!] = \mathsf{trace}(\langle [\![M]\!], [\![N_1]\!]; \mathsf{cell}_{[\![T_1]\!]}, \ldots, [\![N_k]\!]; \mathsf{cell}_{[\![T_k]\!]}\rangle)$.

We may now verify directly that the key reductions of the operational semantics are sound, using the properties of the trace operator and the read-write behaviour of the cell strategy which we have identified (see Laird [2002] for further details).

Proposition 6.2 (Soundness). *If $M \Downarrow$, then $[\![M]\!] = \top$.*

We prove *computational adequacy* for an approximating semantics in which each cell can be accessed a bounded number of times, which implies adequacy for the unbounded system by continuity. Notably, this avoids the logical complexity of arguments, such as those based on reducibility candidates, which are required to show strong normalization for System F (or, indeed, to show that our operational semantics is conservative over β-reduction).

Proposition 6.3. *$[\![M]\!] = \top$ implies $M \Downarrow$.*

Proof. For each n, we define an operational semantics of System $\mathbf{F}_{\mathtt{ref}}$ in which each cell may be (read or write) accessed at most n times—that is, environments are sets of triples (a, M, k), and the rules for declaration, assignment and dereferencing are:

$$\begin{array}{lcl} E[(\lambda x.M)\, N]|\mathcal{E} & \longrightarrow & E[M[a/x]]|\mathcal{E}, (a, N, n) \\ E[\mathtt{set}(a)\, M]|\mathcal{E}, (a, N, k+1) & \longrightarrow & E[()]|\mathcal{E}, (a, M, k) \\ E[a]|\mathcal{E}, (a, M, k+1) & \longrightarrow & E[M]|\mathcal{E}, (a, M, k). \end{array}$$

Write $M \Downarrow^n$ if $M : \mathsf{com}$ (closed) reduces to $()$ in this semantics. Evidently, $M \Downarrow^n$ implies $M \Downarrow$.

We show that reduction with respect to this operational semantics always terminates, by defining a measure on terms as follows:

—$l_n(c) = 1$ for all constants (and variables) c.
—$l_n(\lambda x.M) = l_n(M) + 1$.
—$l_n(M; N) = l_n(M) + l_n(N) + 1$.
—$l_n(M\,N) = l_n(M) + n \cdot l(N)$.
—$l_n(\Lambda X.M) = l_n(M) + 1$.
—$l_n(M\{T\}) = l_n(M) + 1$.

This extends to configurations: $l_n(M|\mathcal{E}) = l_n(M) + \Sigma\{i \cdot l_n(N) \mid (a, N, i) \in \mathcal{E}\}$. It is routine to show that reduction in System $\mathrm{F}^n_{\mathtt{ref}}$ is strictly reducing on l_n—that is, if $M|\mathcal{E} \longrightarrow M'|\mathcal{E}'$ then $l_n(M'|\mathcal{E}') < l_n(M|\mathcal{E})$, and hence there are no infinite reductions.

We then show that this operational semantics is sound, and therefore adequate, with respect to a denotational semantics in which the cell strategy is replaced by its nth approximant—that is, $\Phi^n(\bot)$.

Adequacy of the semantics of System $\mathrm{F}_{\mathtt{ref}}$ now follows. For any term $M : \mathsf{com}$, $[\![M]\!] = \bigcup_{n\in\omega} [\![M]\!]_n$. Hence, if $[\![M]\!] \neq \bot$, there exists n with $[\![M]\!]_n \neq \bot$ and hence $M \Downarrow^n$, and so $M \Downarrow$ as required. □

7. COMPLETENESS: GENERICITY, DEFINABILITY AND FULL ABSTRACTION

What of completeness? Interpretation of System F is not *fully complete*—that is, there are strategies which do not correspond to proofs of second order intuitionistic logic—obvious examples include the empty strategy and denotations of imperative objects such as the reference cell. Is there a simple criterion for excluding such strategies and identifying the (System F) definable elements of the model? If we consider the semantics of second-order implicational multiplicative linear logic (i.e., linear System F) suggested by our symmetric monoidal category of games with well-opened exponentials, all types are interpreted as finite games (and all proofs as *hereditarily total* strategies). However, undecidability of provability in second-order MLL [Lincoln et al. 1997] implies that definability of a total strategy as a proof or pure term is not decidable even in this case.

An instructive example, illustrating the difficulty of identifying the strategies which are definable in pure System F is given in Figure 3 by plays in the strategies denoted by the terms:

$$\text{—}N_1 : \forall X.\forall Y.(((Y \to X \to X) \to Y \to X) \to X)$$
$$= \Lambda X.\Lambda Y.\mathsf{new}\, x^Y.\lambda f^{(Y\to X\to X)\to Y\to X}.(f\ \lambda y^Y z^X.x := y; z)\, x$$
$$\text{—}N_2 : \forall X.((\forall Y.((X \to Y \to X) \to Y \to X)) \to X)$$
$$= \Lambda X.\lambda f^{\forall Y.((X\to Y\to X)\to Y\to X)}.(f\{X \to X\}\,(\lambda x^X.\lambda g^{X\to X}.g\,x))\,\lambda y^X.y$$

which are played over the same basic game (but with a different Q/A-labelling). The plays are concretely the same, although the former corresponds to quintessential general reference behaviour.

7.1. Genericity

A different kind of completeness property, the genericity principle for System F was described by Longo et al. [1993], and shown to be consistent with the equational theory of System F (i.e., models satisfying the principle exist).

Definition 7.1. A semantics of System F satisfies the *genericity principle* at type T if for any terms $M, N : \forall X.S$, $[\![M\{T\}]\!] = [\![N\{T\}]\!]$ implies $[\![M]\!] = [\![N]\!]$. It satisfies the (all-type) genericity principle if it satisfies the genericity principle at each type.

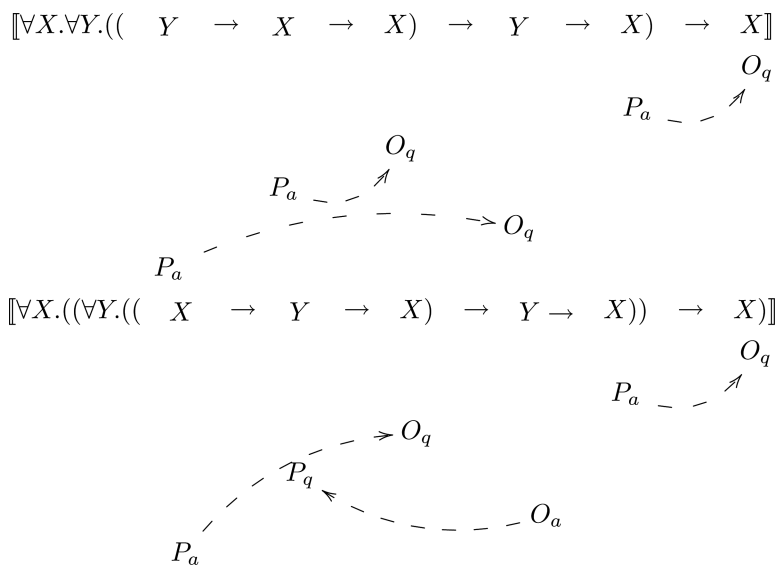


Fig. 3. Typical plays of N_1 and N_2 (Pointers indicate bracketing).

In other words, in a model satisfying the all-type genericity principle, separability of denotations at the universal type $\forall X.T$ is parametric in X. Evidently, genericity in our semantics of System $\mathrm{F}_{\mathtt{ref}}$ implies genericity in the submodel of System F itself.

One possible criticism of the all-type genericity principle is that it is rather brittle. If we include finite product types in System F—in particular, a terminal type $\mathbf{1}$ at which all terms are equated, then genericity implies inconsistency: we have, for example, $(\Lambda X.\lambda x^X.\lambda y^X.x)\{\mathbf{1}\} = (\Lambda X.\lambda x^X.\lambda y^X.y)\{\mathbf{1}\}$, and thus $M : T = ((\Lambda X.\lambda x^X.\lambda y^X.x)\{T\}\, M)\, N = (\Lambda X.\lambda x^X.\lambda y^X.y)\{T\}\, M)\, N = N : T$ for any pair of terms $M, N : T$. (So genericity is inconsistent with the faithful encoding of finite products in System F.)

It is straightforward to characterize the (closed) generic types in our model. Recall that a legal sequence is *complete* if it contains equal numbers of questions and answers—that is, by the well-bracketing condition, every question is closed by a unique answer. Say that a game A is complete if there is at least one nonempty complete legal sequence in A. (So $[\![\forall X.X \to X]\!] = \Sigma$ is complete, but $[\![\forall X.X]\!]$ is not.)

Proposition 7.2. *If $[\![T]\!]$ is complete, then T is generic.*

Proof. Suppose $[\![T]\!]$ contains the complete sequence $m_1 \cdots m_{2k}$. Suppose s is a legal sequence over $[\![\forall X.S]\!]$ (where $\forall X.S$ is closed). We may extend s to a legal sequence s' over $[\![S[T/X]]\!]$ by replacing each 0-hole question $u \in H^+_{[\![S]\!]}(0)$ with $u \cdot m_1$, and each answer $v \in H^-_{[\![S]\!]}(0)$ such that $(u, v) \in \mathsf{Cl}_{[\![S]\!]}(s)$ with the sequence $(v \cdot m_1)(v \cdot m_2)(u \cdot m_2) \cdots (v \cdot m_{2k})(u \cdot m_{2k})$. This is evidently injective on legal sequences, and has the property that $s \in [\![M]\!]$ if and only if $s' \in [\![M\{T\}]\!]$, so if $[\![M\{T\}]\!] = [\![N\{T\}]\!]$, then $[\![M]\!] = [\![N]\!]$ as required. □

Proposition 7.3. *If T (closed) is generic, then $[\![T]\!]$ is complete.*

Proof. Consider the terms of type $\forall X.\forall Y.(\forall Z.((((Y \to Y \to Z) \to X) \to X) \to Z) \to Y)$:

$$—M_1 = \Lambda X.\Lambda Y.\lambda f^{\forall Z.(((Y\to Y\to Z)\to X)\to X)\to Z}.\, f\{Y\}\ \lambda g^{(Y\to Y\to Y)\to X}.g\ \lambda x^Y.\lambda y^Y.x.$$
$$—M_2 = \Lambda X.\Lambda Y.\lambda f^{\forall Z.(((Y\to Y\to Z)\to X)\to X)\to Z}.\, f\{Y\}\ \lambda g^{(Y\to Y\to Y)\to X}.g\ \lambda x^Y.\lambda y^Y.y.$$

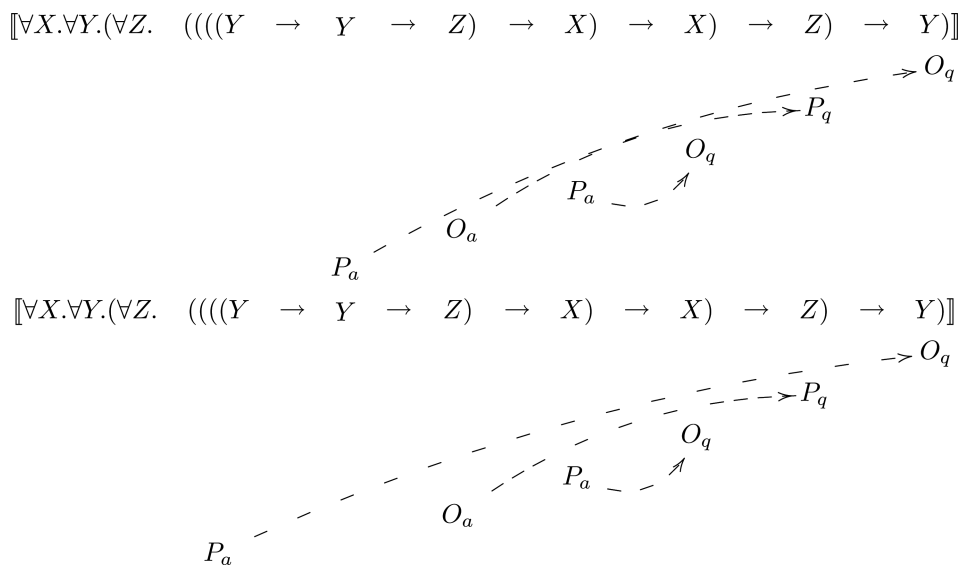


Fig. 4. Typical plays of $[\![M_1]\!]$ and $[\![M_2]\!]$.

Then $[\![M_1]\!] \neq [\![M_2]\!]$ as they are separated by the plays depicted in Figure 4. If $[\![M_1\{T\}]\!] \neq [\![M_2\{T\}]\!]$, then there exist well-bracketed plays $s_1 \in [\![M_1\{T\}]\!], s_2 \in [\![M_2\{T\}]\!]$ which restrict to the depicted plays. The restriction of s_1, s_2 to (each copy of) $[\![T]\!]$ must be complete and non-empty, and therefore $[\![T]\!]$ is complete. □

Thus, we have a simple and exhaustive characterization of the generic closed types in our model. Moreover, we may use it as the basis for an all-type generic model via a simple construction which adds an answer to the initial question to each type object, making it a complete game. We may describe this construction syntactically, as a translation $\overline{(\_)}$ from System F into itself—providing a concise and clear way to formalise it and establish soundness. Translation of types is as follows:

—$\overline{X} = X$
—$\overline{S \to T} = \overline{S} \to \overline{T}$
—$\overline{\forall X.T} = \forall X.(X \to \overline{T})$.

Since universal quantification now introduces an answer to any question corresponding to a quantified type variable, we have:

PROPOSITION 7.4. *For any type T, $[\![\overline{T}]\!]$ is complete.*

To define the translation on terms, for each type $X_1, \ldots, X_n \vdash T$, we define a term $X_1, \ldots, X_n; x_1 : X_1, \ldots, x_n : X_n \vdash K_T : \overline{T}$ as follows:

—$K_{X_i} = x_i$.
—$K_{S \to T} = \lambda y^{\overline{S}}.K_T$.
—$K_{\forall X_i.T} = \Lambda X_i.\lambda x_i^{X_i}.K_T$.

By induction over this definition, we have:

LEMMA 7.5. *For any type $T(X_i)$, $K_{T[S/X_i]} = (K_T[\overline{S}/X_i])[K_S/x_i]$.*

We may now translate terms $X_1, \ldots, X_n; \Gamma \vdash M : T$ of System F as terms $X_1, \ldots, X_n; x_1 : X_1, \ldots x_n : X_n, \overline{\Gamma} \vdash \overline{M} : \overline{T}$ as follows:

—$\overline{y : T} = y : \overline{T}$.
—$\overline{\lambda y^S.M : S \to T} = \lambda y^{\overline{S}}.\overline{M}$.
—$\overline{M\,N} = \overline{M}\,\overline{N}$
—$\overline{\Lambda X_i.M} = \Lambda X_i.\lambda x_i^{X_i}.\overline{M}$
—$\overline{M\{T\}} = \overline{M}\{\overline{T}\}\,K_T$

LEMMA 7.6. *For any term M,* $\overline{M}[\overline{T}/X_i][K_T/x_i] = \overline{M[T/X_i]}$.

PROOF. By induction on term length, using Lemma 7.5 □

LEMMA 7.7. *The translation* $\overline{(\_)}$ *is sound with respect to* $\beta\eta$*-equality.*

PROOF. The key cases are:

—$\overline{(\Lambda X_i.M)\{T\}} = (\Lambda X_i.\lambda x_i^{X_i}.\overline{M})\{\overline{T}\}\,K_T =_{\beta\eta} \overline{M}[\overline{T}/X_i][K_T/x_i] = \overline{M[T/X_i]}$.
—$\overline{\Lambda X_i.(M\{X_i\})} = (\Lambda X_i.\lambda x_i^{X_i}.(\overline{M}\{X_i\}\,x_i)) =_{\beta\eta} \overline{M}$ as required. □

This construction yields an interpretation in which all closed types denote complete games. However, since the interpretation of type instantiation has changed (it now involves composition with the term K_T), we need to check that this does not affect genericity.

Define an *explicitly complete* context game to be a context game A, together with a function $\kappa_A : \mathsf{Qn}_A \to \mathsf{Ans}_A$ such that $q \rhd \kappa_A(q)$. A *strategy* on A is *explicitly complete* with respect to κ if for all $smn \in \sigma$, m is an explicit answer if and only if n is an explicit answer. (Note that well-bracketing implies that the response of σ to an explicit answer is therefore completely determined.) We construct a mapping $\kappa_{\Theta,T} : \mathsf{Qn}_{[\![\Theta \vdash \overline{T}]\!]} \to \mathsf{Ans}_{[\![\Theta \vdash \overline{T}]\!]}$ for each System F type (sending each question to the answer introduced by $\overline{(\_)}$-translation) by induction on T, and verify the following by induction: (Alternatively, we may describe the semantics of System F in the category of explicitly complete games and strategies *ab initio*.)

LEMMA 7.8. *For any term M,* $[\![\overline{M}]\!]$ *is an explicitly complete strategy.*

We may now adapt the proof of Proposition 7.2 to establish genericity for the modified model of System $\mathrm{F}_{\mathtt{ref}}$.

THEOREM 7.9 (GENERICITY). *For any terms* $M, N : \forall X.S$*, and (closed) type* T*,* $[\![\overline{M\{T\}}]\!] = [\![\overline{N\{T\}}]\!]$ *implies* $[\![\overline{M}]\!] = [\![\overline{N}]\!]$.

PROOF. It suffices to prove the result for terms of closed type $\forall X.S$ (since we may thus show that genericity holds for any sequence of instantiations to a closed type). Suppose $[\![\overline{M}]\!] \neq [\![\overline{N}]\!]$. Let s be the shortest sequence such $s \in [\![\overline{M}]\!]$ and $s \notin [\![\overline{N}]\!]$. Then, s does not contain the explicit answer to the initial question, since it would otherwise have the form tmn, where $t \in [\![\overline{N}]\!]$ and m and n are explicit answers and so $s = tmn \in [\![\overline{N}]\!]$. Because it is complete, instantiating with $\overline{T}$ yields a sequence s' on $[\![\overline{M}\{\overline{T}\}]\!] : [\![\overline{T}]\!] \to [\![\overline{S[T/X]}]\!]$ as in Proposition 7.2. Since s does not contain the explicit answer to the initial question, s' does not contain any moves in $\overline{T}$ and so $s' \upharpoonright [\![\overline{S[T/X]}]\!]$ is in $[\![\overline{M\{T\}}]\!] = [\![\overline{M}\{\overline{T}\}\,K_T]\!]$ but not in $[\![\overline{N\{T\}}]\!] = [\![\overline{N}\{\overline{T}\}\,K_T]\!]$ as required. □

Note that we have achieved genericity by adding more "junk" (undefinable elements) to our model. In the following, we return to our original interpretation of System $\mathrm{F}_{\mathtt{ref}}$.

7.2. Definability and Full Abstraction

We now prove that every finite strategy in our model of System $\mathrm{F}_{\mathtt{ref}}$ is definable as a term, using a decomposition argument based on the one given in Laird [2002] for a simply-typed language with general references. The principal question to be addressed in extending this decomposition to polymorphic types is: how to instantiate universal types which occur negatively? Strategies do not carry direct information about such instantiations. We show that it is sufficient to always instantiate with the type com. Since such instantiation typically increases sequence length, we first need to define a measure on finite strategies which allows it to be performed as part of the inductive decomposition. For each type T, let $\mathsf{com?}(T)$ be the set of questions in the game $[\![T]\!]$ which come from subgames denoted by com—that is, $\mathsf{com?}(\mathsf{com})$ is the unique question in Σ, $\mathsf{com?}(X) = \varnothing$, $\mathsf{com?}(\forall X.T) = \mathsf{com?}(T)$ and $\mathsf{com?}(S \to T) = \mathsf{com?}(S) \uplus \mathsf{com?}(T)$. Define the *noncommand Player question count* of a legal sequence s on $[\![T]\!]$ to be the number of Player questions in s which are not in $\mathsf{com?}(T)$, and let $\#_T(s)$ be the pair $\langle m, n\rangle$, where m is the noncommand question count of s and n is the length of s. We order these pairs lexicographically. This measure extends to finite strategies $\sigma : [\![T]\!]$: let $\#_T(\sigma)$ be the maximum of $\{\#_T(s) \mid s \in \sigma\}$.

A strategy $\sigma : [\![T]\!]$ in $\mathcal{G}$ is *definable* if there is a term $M_\sigma : T$ such that $[\![M_\sigma]\!] = \sigma$. Say that σ is (definability) reducible to the finite set of strategies $\{\sigma_i : [\![T_i]\!] \mid i \in I\}$ if $\#_{T_i}(\sigma_i) \leq \#_T(\sigma)$ for each i, and definability of each σ_i implies definability of σ. σ is *strictly* reducible to $\{\sigma_i : [\![T_i]\!] \mid i \in I\}$ if in addition $\#_{T_i}(\sigma_i) < \#_T(\sigma)$ for each $i \in I$.

To simplify the proof, we work with types which are *canonical*—that is, given by the grammar:

$$C, C' ::= X \mid \mathsf{com} \mid (\forall X_1 \cdots \forall X_n.C) \to C'$$

A simple induction on types yields:

LEMMA 7.10. *For any type T, there exists a canonical type C such that T is definably isomorphic to $\forall X_1 \ldots X_n.C$ (i.e. there are terms $M : T \to \forall X_1 \ldots \forall X_n.C$, $N : \forall X_1 \ldots \forall X_n.C \to T$ denoting an isomorphism.)*

Thus:

LEMMA 7.11. *Any finite strategy $\sigma : [\![T]\!]$ is reducible to a strategy $\sigma' : [\![C]\!]$ such that C is canonical.*

Hence, it suffices to prove definability at canonical types. Any such type which is nonatomic is a function-type $S \to T$. We shall say that a strategy on $[\![S \to T]\!]$ is *linear* if it is (the currying of) the dereliction of a strict morphism $\tau : [\![S]\!] \to [\![T]\!]$. In other words, the first Player move by σ is in $![\![S]\!]$, and σ opens only one "thread" of $![\![S]\!]$.

LEMMA 7.12. *Any linear strategy $\sigma : [\![\forall X.S \to T]\!]$ is reducible to a linear strategy $\sigma' : [\![S[\mathsf{com}/X] \to T]\!]$.*

PROOF. Given a legal sequence s in $!(\forall_0 A) \multimap B$, we define a (legal) sequence s' on $A[\Sigma] \to B$ as follows: for each pair $(u, v) \in \mathsf{Cl}(s)$ with $u, v \in H_A(0)$, add the moves $(u \cdot 1)(v \cdot 1)$ after v, where 1 is the answer move in Σ. Given $\sigma : [\![\forall X.S \to T]\!]$, let σ' be the strategy $\{t \in L_{![\![S]\!][\Sigma] \to [\![T]\!]} \mid \exists s \in \sigma.t \sqsubseteq^E s'\}$. Then:

—$\#_{S[\mathsf{com}/X] \to T}(\sigma') \leq \#_{\forall X.S \to T}(\sigma)$, since for any sequence $s \in \sigma$, either $s' = s$, or s' has strictly fewer noncommand Player questions.
—If σ' is definable as the term $M : S[\mathsf{com}/X] \to T$, then σ is definable as the term $\lambda y^{\forall X.S}.M\,(y\{\mathsf{com}\})$. □

LEMMA 7.13. *Any linear strategy* $\sigma : [\![(\vec{S} \to \mathsf{com}) \to \vec{T} \to X]\!]$ *is reducible to a linear strategy* $\sigma' : [\![(\vec{S} \to X \to X) \to \vec{T} \to X]\!]$.

PROOF. Given a legal sequence $s = oqt$ in $\sigma : [\![(\vec{S} \to \mathsf{com}) \to \vec{T} \to X]\!]$ (where q is the question move in com), form the legal sequence $s' = oat'$ on $[\![\vec{S} \to X \to X]\!] \to [\![\vec{T} \to X]\!]$ by replacing q with the initial move a in $X \to X$, and its answer in t with the corresponding move in $X \to X$ enabled by q. (Note that this preserves well-bracketing.) Thus, we may define $\sigma' : [\![\vec{S} \to X \to X]\!] \to [\![\vec{T} \to X]\!] = \{s' \mid s \in \sigma\} \cup \{\varepsilon\}$ such that:

—$\#(\sigma) = \#(\sigma')$ since this relabelling of moves does not change sequence length, or number of noncommand player questions.
—If σ' is definable as the term $M : (\vec{S} \to X \to X) \to \vec{T} \to X$, then σ is definable as the term $\lambda y^{\vec{S} \to \mathsf{com}}.M\, \lambda \vec{z}^{\vec{S}}.(y\, \vec{z}); (\lambda a^X.a)$. □

Given types $T, S_1, \ldots, S_n$, write $(T \to \vec{S}) \to X$ for $(T \to S_1) \to \cdots \to (T \to S_n) \to X$.

LEMMA 7.14. *Any linear strategy* $\sigma : [\![(\vec{S} \to X) \to (T \to R)]\!]$ *is reducible to a linear strategy on* $[\![((T \to \vec{S}) \to X) \to R]\!]$

PROOF. Any sequence $s \in \sigma$ of length greater than 2 has the form $oam_i s$, where m_i is an Opponent move in some $[\![S_i]\!]$. We may rewrite s to a sequence s' on $[\![((T \to \vec{S}) \to X) \to R]\!]$ by replacing moves in $![\![T]\!]$ with the corresponding moves in (the first thread of) $![\![T \to S_i]\!]$, and define $\sigma' = \{s' \mid s \in \sigma\} \cup \{\varepsilon, oa\}$. Then, σ is reducible to σ' as:

—$\#(\sigma) = \#(\sigma')$.
—If σ' is definable as a term $M : ((T \to \vec{S}) \to X) \to R$, then σ is definable as the term $\lambda x^{\vec{S} \to X}.\lambda y^T.M\, \lambda \vec{z}^{T \to \vec{S}}.x\, (z_1\, y) \cdots (z_n\, y)$. □

An *assignment strategy* $\sigma : [\![(\vec{S} \to \mathsf{com}) \to T]\!]$ is a linear strategy in which every sequence of length greater than 2 has the form $oqas$—that is, the initial Player question in $[\![\vec{S} \to \mathsf{com}]\!]$ is answered immediately.

LEMMA 7.15. *Any linear strategy* $\sigma : [\![(\vec{T} \to X) \to X]\!]$ *is reducible to a set of assignment strategies* $\{\sigma_i : [\![(\vec{T} \to \mathsf{com}) \to T_i]\!] \mid i \leq n\}$.

PROOF

—If σ is nonempty, then there exists a strategy $\sigma' : I \to ![\![T_1]\!] \otimes \cdots \otimes ![\![T_n]\!]$ such that $\sigma = (\mathsf{id}_{[\![\vec{T} \to X]\!]} \otimes \sigma'); \mathsf{app}$.
—By sequential decomposability of $\otimes$, and the minimal invariance property of the cofree commutative comonoid, $![\![T_1]\!] \otimes \cdots \otimes ![\![T_n]\!]$ is a Cartesian product of $[\![T_i]\!] \oslash (![\![T_1]\!] \otimes \cdots \otimes ![\![T_n]\!]) : i \leq n$, and thus σ' is a tuple of strategies $\sigma_i : [\![T_i]\!] \oslash ![\![T_1]\!] \otimes \cdots \otimes ![\![T_n]\!]$.
—For each i, we obtain an assignment strategy $\widehat{\sigma_i} : [\![(\vec{T} \to \mathsf{com}) \to T_i]\!]$ by adding a pair of moves qa in com to each (nonempty) sequence in σ_i, immediately after the opening move.

σ reduces to $\{\widehat{\sigma_i} \mid i \leq n\}$ since:

—Decomposition of σ to σ' removes two moves, decomposition of σ' to σ_i is nonincreasing on $\#(\_)$, and decomposition of σ_i to $\widehat{\sigma_i}$ adds two moves (which are not noncommand Player questions).

—If each $\widehat{\sigma_i}$ is definable as a term $M_i : (\vec{T} \to \mathsf{com}) \to T_i$, then σ is definable as the term:

$$\lambda f^{\vec{T} \to X}.\mathsf{new}\ \vec{w}^{\vec{T}}.w_1 := (M_1\, \lambda \vec{x}^{\vec{T}}.\vec{w} := \vec{x}); \cdots ; w_n := (M_n\, \lambda \vec{x}^{\vec{T}}.\vec{w} := \vec{x}); f\ w_1 \cdots w_n. \quad \square$$

Suppose $T = T_1 \to \cdots \to T_n \to X$.

LEMMA 7.16. *Any assignment strategy* $\sigma : [\![(\vec{S} \to \mathsf{com}) \to T]\!]$ *is strictly reducible to a linear strategy* $\sigma' : [\![(\vec{S} \to T_i) \to T]\!]$ *for some* $i \leq n$.

PROOF. By definition, any sequence in σ of length greater than 2 has the form $oqam_i s$, where m_i is an opening move in some $![\![T_i]\!]$. Thus, we obtain a strategy $\sigma' : [\![(\vec{S} \to T_i) \to T]\!]$ by deleting qa, and replacing all moves hereditarily enabled by m_i with the corresponding moves in $[\![\vec{S} \to T_i]\!]$.

—The decomposition of σ to σ' strictly reduces the length of sequences in σ as it removes two moves.
—If σ' is definable as a term $M : (\vec{S} \to T_i) \to X$, then σ is definable as the term:

$$\lambda f^{\vec{S} \to \mathsf{com}}.\lambda \vec{y}^{\vec{T}}.\mathsf{new}\ \vec{x}^{\vec{S}}.(f\ \vec{x}); M\,(\lambda \vec{z}^{\vec{S}}.(x_1 := z_1); \cdots ; (x_k := z_k); y_i)\, y_1 \cdots y_n. \quad \square$$

PROPOSITION 7.17. *For any canonical type* T*, every finite linear strategy* $\sigma : [\![T]\!]$ *is the denotation of a term of System* $F_{\mathtt{ref}}$.

PROOF. By induction on the measure $\#_T(\_)$. At the base case, the empty strategy at type T is the denotation of the divergent term at T. If σ is nonempty then:

(1) If T has the form $(\vec{S} \to X) \to X$ then we may apply Lemmas 7.15 and 7.16 to strictly reduce σ to a family of linear strategies on canonical types.
(2) If T has the form $(\vec{S} \to X) \to \vec{R} \to X$ then repeated application of Lemma 7.14 reduces to Case 1.
(3) If T has the form $(\vec{S} \to X) \to \vec{R} \to \mathsf{com}$, then the isomorphism $\mathsf{com} \cong \forall X.(X \to X)$ reduces to Case 2.
(4) If T has the form $(\vec{S} \to \mathsf{com}) \to R$, then Lemma 7.13 reduces to Case 1, Case 2 or Case 3.
(5) If T has the form $\forall \vec{X}.S \to R$, then repeated application of Lemma 7.12 reduces to Case 1, Case 2, Case 3 or Case 4. $\square$

Note that definability of linear strategies on $[\![(T \to X) \to X]\!]$ implies definability of arbitrary strategies on T.

We may now give a simple and direct characterization of contextual equivalence and approximation in our model. For any strategy σ, let $\mathsf{comp}(\sigma)$ be the set of complete sequences in σ.

THEOREM 7.18 (FULL ABSTRACTION). *For any (closed) terms* M, N*,* $M \lesssim N$ *if and only if* $\mathsf{comp}([\![M]\!]) \subseteq \mathsf{comp}([\![N]\!])$.

PROOF. Assuming M, N are closed terms of closed type T, if $M \not\lesssim N$, then there exists $C[\_] : \mathsf{com}$ such that $C[M] \Downarrow$ and $C[N] \not\Downarrow$. Thus, there exists $s \in [\![M]\!]$ such that $qsa \in [\![\lambda x^T.C[x]]\!]$ and $s \notin [\![N]\!]$. By well-bracketing s is an interleaving of complete sequences, and thus $\mathsf{comp}([\![M]\!]) \not\subseteq \mathsf{comp}([\![N]\!])$.

Conversely, if $\mathsf{comp}([\![M]\!]) \not\subseteq \mathsf{comp}([\![N]\!])$, then there exists a complete sequence $s \in [\![M]\!]$ such that $s \notin [\![N]\!]$ which we may extend to the finite linear strategy on $[\![T \to \mathsf{com}]\!]$ consisting of even prefixes of qsa. This is definable as a term $L : T \to \mathsf{com}$ such that $L\,M \Downarrow$ and $L\,N \not\Downarrow$. $\square$

8. CONCLUSIONS AND FURTHER DIRECTIONS

The major contribution of this article has been to lay the foundations for an *intensional semantics* for polymorphism and genericity—that is, a model in which the meaning of a polymorphic program is given formally by a blueprint for computing it at any instantiating type, rather than a direct representation of the set of such instances. This is closer to the way that programmers actually use and think about genericity. We have shown that the structure required to represent generic programs (question and answer labellings and relations) is both surprisingly simple, and closely related to existing concepts in game semantics. Interestingly, it does not appear to be consistent with only static binding of variables (innocence and visibility [Hyland and Ong 2000]).

Further work proceeding from these foundations includes:

—*Computational Effects.* Game models for a wide variety of languages with different computational effects (and none), have been identified, by imposing different constraints on strategies. The extent to which these results extend to polymorphic types is an open question—we have argued that there are no simple constraints on our model yielding a fully complete model of System F itself, but many other possibilities exist.

—*Polymorphism for Call-by-Value.* Our semantics extends readily to polymorphic *computation types* in languages such as *call-by-push-value* [Levy 2004]. To describe polymorphic *value types* requires a different approach in which copycat links are captured by explicit pointers [Laird 2010b]. This leaves open the problem of finding an approach which embraces both forms of polymorphism.

—*Model Checking.* Since our model is rather concrete (and, in particular, effectively presentable) it may be possible to apply the methods of *algorithmic game semantics* [Ghica and McCusker 2003] to decide questions of program equivalence for suitably defined fragments of System $\mathrm{F}_{\mathtt{ref}}$ (which is very expressive even at simple types).

—Subtyping: A further goal is extension to a semantics of subtype polymorphism, as in the extension of System F to $\mathrm{F}_{<}$ [Cardelli et al. 1994], working towards a full semantic account of object-oriented polymorphism and inheritance.

ACKNOWLEDGMENTS

The author wishes to thank Pierre Clairambault for useful discussions and the anonymous referees for their contributions to this work.

REFERENCES

Abramsky, S., Honda, K., and McCusker, G. 1998. A fully abstract games semantics for general references. In *Proceedings of the 13th Annual Symposium on Logic In Computer Science (LICS'98)*. IEEE Press.

Abramsky, S. and Jagadeesan, R. 1994. Games and full completeness for multiplicative linear logic. *J. Symb. Log. 59*, 543–574.

Abramsky, S. and Jagadeesan, R. 2004. A game semantics for generic polymorphism. *Ann. Pure Appl. Logic 133,* 1, 3–37.

Abramsky, S., Jagadeesan, R., and Malacaria, P. 2000. Full abstraction for PCF. *Inf. Computat. 163*, 409–470.

Cardelli, L., Mitchell, J. C., Martini, S., and Scedrov, A. 1994. An extension of System F with subtyping. *Inf. Computat. 109,* 1–2, 4–56.

Clairambault, P. 2009. Least and greatest fixpoints in game semantics. In *Proceedings of FOSSACS '09*. Lecture Notes in Computer Science, vol. 5504, Springer.

de Lataillade, J. 2009. Dinatural terms in System F. In *Proceedings of the 24th Annual Symposium on Logic in Computer Science (LICS '09)*. IEEE Press.

Ghica, D. and McCusker, G. 2003. The regular language semantics of second-order Idealised Algol. *Theoreti. Comput. Sci. 309*, 469–502.

Girard, J.-Y. 1972. Interprétation functionelle et elimination des coupures de l'arithmètique d'ordre supérieur. Ph.D. thesis, Université Paris VII.

HUGHES, D. 1997. Games and definability for System F. In *Proceedings of the 12th International Syposium on Logic in Computer Science (LICS'97)*. IEEE Computer Society Press.

HYLAND, J. M. E. AND ONG, C.-H. L. 2000. On full abstraction for PCF: I, II and III. *Inf. Computat. 163*, 285–408.

LAIRD, J. 2002. A categorical semantics of higher-order store. In *Proceedings of CTCS '02*. ENTCS. Elsevier.

LAIRD, J. 2010a. Game semantics for a polymorphic programming language. In *Proceedings of LICS '10*. IEEE Press.

LAIRD, J. 2010b. Game semantics of call-by-value polymorphism. In *Proceedings of ICALP '10*. Lecture Notes in Computer Science, vol. 6198, Springer-Verlag.

LEVY, P. B. 2004. *Call-By-Push-Value*. Semantic Structures in Computation. Kluwer.

LINCOLN, P. D., SCEDROV, A., AND SHANKAR, N. 1997. Decision problems for second order linear logic. In *Proceedings of LISM '97*, M. D. Chiara, Ed., Kluwer.

LONGO, G., MILSTED, K., AND SOLOVIEV, S. 1993. The genericity theorem and parametricity in the polymorphic λ-calculus. *Theoreti. Comput. Sci. 121,* 1&2, 323–349.

MALHERBE, O., SCOTT, P. J., AND SELINGER, P. 2012. Partially traced categories. *J. Pure Appl. Alg. 216*, 12, 2563–2585, DOI: 10.1016/j.jpaa.2012.03.026.

MCCUSKER, G. 1996. Games and full abstraction for a functional metalanguage with recursive types. Ph.D. dissertation, Imperial College London. Published by Cambridge University Press.

MØGELBERG, R. E. AND SIMPSON, A. 2009. Relational parametricity for computational effects. *Log. Meth. Comput. Sci. 5,* 3.

PITTS, A. 1988. Polymorphism is set-theoretic constructively. In *Proceedings of CTCS '88*, D. Pitt, Ed., Lecture Notes in Computer Science, vol. 283, Springer.

REYNOLDS, J. C. 1974. Towards a theory of type structure. In *Proceedings of the Programming Symposium, Paris 1974*. Lecture Notes in Computer Science, vol. 19, Springer.

REYNOLDS, J. C. 1983. Types, abstraction and parametric polymorphism. *Inf. Proc. 83*, 513–523.

SEELY, R. A. G. 1987. Categorical semantics for higher-order polymorphic lambda-calculus. *J. Symb. Logic 52,* 4, 969–989.

Received August 2011; revised May 2012, November 2012, January 2013, and April 2013; accepted April 2013